

**UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
PROJETO DE GRADUAÇÃO**



CAIO BORGIO

**ESTUDO APROFUNDADO E DESENVOLVIMENTO DE UM
SIMULADOR DE CIRCUITOS ELÉTRICOS PARA
ANÁLISES TRANSIENTES**

VITÓRIA
2021

CAIO BORGIO

**ESTUDO APROFUNDADO E DESENVOLVIMENTO DE UM
SIMULADOR DE CIRCUITOS ELÉTRICOS PARA ANÁLISES
TRANSIENTES**

Parte manuscrita do Projeto de Graduação do aluno **Caio Borgo**, apresentada ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Orientador: Prof. Dr. André Ferreira

VITÓRIA
2021

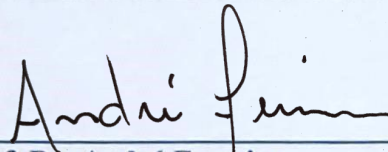
CAIO BORGIO

**ESTUDO APROFUNDADO E DESENVOLVIMENTO DE UM
SIMULADOR DE CIRCUITOS ELÉTRICOS PARA ANÁLISES
TRANSIENTES**

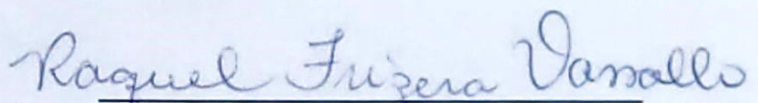
Parte manuscrita do Projeto de Graduação do aluno **Caio Borgo**, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Aprovada em 08 de outubro de 2021.

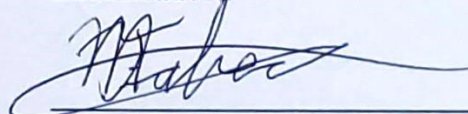
COMISSÃO EXAMINADORA:



Prof. Dr. André Ferreira
Universidade Federal do Espírito Santo
Orientador



Profa. Dra. Raquel Frizera Vassallo
Universidade Federal do Espírito Santo
Examinadora



Eng. Me. Menno Jan Faber
2Solve Engenharia e Tecnologia Ltda.
Examinador

Aos meus pais, Giovanni e Ana Cláudia, pelo apoio incondicional.

AGRADECIMENTOS

Primeiramente, agradeço a Deus por me proporcionar saúde e sabedoria.

Agradeço à minha família, em especial aos meus pais e meu irmão, que me apoiaram e incentivaram em todos os momentos da minha vida com afeto, conselhos e positividade.

Agradeço aos meus professores, em especial ao Prof. Dr. André Ferreira, por sua orientação, apoio e disponibilidade durante o curso de graduação.

Agradeço aos meus amigos, que me acompanharam durante a jornada que é o curso de Engenharia Elétrica.

Por fim, agradeço à Universidade Federal do Espírito Santo e todos os seus funcionários, pelo ensino público, gratuito e de qualidade.

RESUMO

Desde a invenção dos primeiros computadores, simulações computacionais são desenvolvidas como uma ferramenta auxiliar a projetos. Por proporcionarem uma forma pouco custosa e relativamente rápida de prever o comportamento de sistemas, simuladores se tornaram essenciais em empreendimentos complexos, onde testes práticos podem trazer consequências negativas tanto em termos econômicos quanto na segurança de trabalhadores. No contexto educacional, simuladores podem facilitar o aprendizado de estudantes e o ensino para professores, pois permitem a rápida modificação de parâmetros e exigem poucos equipamentos para sua aplicação. Com base na importância desse tipo de ferramenta, o projeto teve como objetivo desenvolver um simulador de circuitos elétricos, incluindo o estudo e implementação de métodos matemáticos associados e a criação de uma interface gráfica. Para isso, realizou-se uma análise do funcionamento de simuladores já existentes, especialmente daqueles baseados em SPICE (*Simulation Program with Integrated Circuit Emphasis*). A versão final da aplicação consegue simular circuitos que contêm componentes básicos como fontes de tensão e corrente, resistores, capacitores, diodos, dentre outros. Seu desenvolvimento foi voltado para a concepção de uma ferramenta de uso simples e intuitivo, porém com precisão satisfatória em comparação a produtos já consolidados no mercado.

Palavras-chave: Simulações computacionais. SPICE. Circuitos elétricos. Método de análise nodal modificada.

ABSTRACT

Since the invention of the first computers, computer simulations have been developed as an auxiliary tool for projects. Since simulators provide an inexpensive and relatively fast way to predict the behaviour of systems, they have become essential in complex tasks, in which practical experiments may lead to negative consequences in both economic and safety terms. In educational contexts, simulators can facilitate the learning process for students, as well as assist teachers with lectures, as these tools allow for quick modification of parameters and require little equipment to be used. Based on the importance of such applications, the project aimed to develop an electric circuit simulator, including the research and implementation of the associated mathematical methods and the creation of a graphical user interface. For this purpose, an analysis of existing simulators was carried out, especially of those based on SPICE (Simulation Program with Integrated Circuit Emphasis). The final version of the application can simulate circuits containing voltage and current sources, resistors, capacitors, diodes, and others. Its development was aimed at designing a tool that is simple and intuitive, but with satisfactory precision in comparison to other consolidated products in the market.

Keywords: Computer simulations. SPICE. Electric Circuits. Modified Nodal Analysis.

LISTA DE FIGURAS

Figura 1 – Exemplo de <i>netlist</i> do SPICE	24
Figura 2 – Circuito de exemplo para o método de análise nodal modificada.....	27
Figura 3 – Componentes lineares básicos.....	30
Figura 4 – Fontes dependentes de tensão	33
Figura 5 – Procedimento para fontes controladas por corrente	34
Figura 6 – Algoritmo de redução gradual da condutância auxiliar	40
Figura 7 – Circuito linear equivalente do diodo	43
Figura 8 – Modelo de Ebers-Moll para o BJT do tipo NPN.....	43
Figura 9 – Modelo para o BJT NPN baseado em componentes lineares.....	46
Figura 10 – Modelo de Ebers-Moll para o BJT do tipo PNP	47
Figura 11 – Modelo para o BJT PNP baseado em componentes lineares	47
Figura 12 – Modelo para o MOSFET de canal tipo N baseado em componentes lineares	48
Figura 13 – Modelo para o MOSFET de canal tipo P baseado em componentes lineares	50
Figura 14 – Circuito linear equivalente para o capacitor.....	52
Figura 15 – Circuito linear equivalente para o indutor.....	53
Figura 16 – Tempo de execução do algoritmo <i>Neighbor-Joining</i> para diferentes linguagens de programação.....	56
Figura 17 – Diagrama do algoritmo do simulador.....	58
Figura 18 – Arquivo de texto para retificador de meia onda.....	62
Figura 19 – Janela inicial do simulador	66
Figura 20 – Descrição dos elementos básicos da interface do simulador.....	67
Figura 21 – Descrição de elementos usados para a montagem de diagramas	67
Figura 22 – Janela de resultados da simulação	68
Figura 23 – Elementos da janela de resultados de simulação.....	68
Figura 24 – Exemplos configuração de parâmetros	69
Figura 25 – Diagrama e resultados para o primeiro circuito no simulador desenvolvido.....	70
Figura 26 – Diagrama e resultados para o primeiro circuito no LTspice	70
Figura 27 – Diagrama e resultados para o primeiro circuito no QUCS.....	71
Figura 28 – Diagrama e resultados para o segundo circuito no simulador desenvolvido	74
Figura 29 – Diagrama e resultados para o segundo circuito no LTspice.....	74
Figura 30 – Diagrama e resultados para o segundo circuito retificador no QUCS	74
Figura 31 – Diagrama e resultados para o terceiro circuito no simulador desenvolvido	78

Figura 32 – Diagrama e resultados para o terceiro circuito no LTspice.....	78
Figura 33 – Diagrama e resultados para o terceiro circuito no QUCS	79
Figura 34 – Diagrama e resultados para o circuito seguidor de emissor no <i>Falstad Circuit Simulator</i>	82
Figura 35 – Modelo do transistor BJT NPN usado pelo LTspice.....	83

LISTA DE GRÁFICOS

Gráfico 1 – Curva IxV do diodo e reta que a aproxima no ponto (V_{D0} , I_{D0})	41
Gráfico 2 – Erros obtidos para a tensão sobre o capacitor no primeiro circuito	71
Gráfico 3 – Erros obtidos para a tensão sobre o indutor no primeiro circuito.....	72
Gráfico 4 – Erros obtidos para a tensão sobre o resistor no primeiro circuito	72
Gráfico 5 – Erros obtidos para a tensão sobre a fonte de tensão no segundo circuito.....	75
Gráfico 6 – Erros obtidos para a tensão sobre a carga no segundo circuito	75
Gráfico 7 – Erros obtidos para a tensão sobre o diodo no segundo circuito	76
Gráfico 8 – Erros obtidos para a corrente sobre a carga e capacitor no segundo circuito	76
Gráfico 9 – Erros obtidos para a corrente sobre o diodo no segundo circuito.....	77
Gráfico 10 – Erros obtidos para a tensão de entrada no terceiro circuito.....	79
Gráfico 11 – Erros obtidos para a tensão de saída no terceiro circuito	80
Gráfico 12 – Ampliação do gráfico dos erros obtidos para a tensão sobre a carga no terceiro circuito	80

LISTA DE TABELAS

Tabela 1 – Termos que devem ser somados aos elementos da matriz G , para um resistor	31
Tabela 2 – Termos que devem ser somados aos elementos da matriz i , para uma fonte independente de corrente	31
Tabela 3 – Termos que devem ser somados aos elementos da matriz B , para uma fonte independente de tensão	31
Tabela 4 – Termos que devem ser somados aos elementos da matriz C , para uma fonte independente de tensão	32
Tabela 5 – Termos que devem ser somados aos elementos da matriz e , para uma fonte independente de tensão	32
Tabela 6 – Termos que devem ser somados aos elementos da matriz A , para uma FCCT	33
Tabela 7 – Termos que devem ser somados aos elementos da matriz A , para uma FTCT ...	34
Tabela 8 – Termos que devem ser somados aos elementos da matriz A , para uma FCCC....	35
Tabela 9 – Termos que devem ser somados aos elementos da matriz A , para uma FTCC ...	35
Tabela 10 – Termos que devem ser somados aos elementos da matriz A , para um BJT do tipo NPN	45
Tabela 11 – Termos que devem ser somados aos elementos da matriz z , para um BJT do tipo NPN	45
Tabela 12 – Variáveis de erro calculadas em relação ao LTspice para o primeiro circuito ...	73
Tabela 13 – Variáveis de erro calculadas em relação ao QUCS para o primeiro circuito	73
Tabela 14 – Variáveis de erro calculadas em relação ao LTspice para o segundo circuito ...	77
Tabela 15 – Variáveis de erro calculadas em relação ao QUCS para o segundo circuito	77
Tabela 16 – Variáveis de erro calculadas para o circuito seguidor de emissor em relação ao LTspice	81
Tabela 17 – Variáveis de erro calculadas para o circuito seguidor de emissor em relação ao QUCS.....	81

LISTA DE ABREVIATURAS E SIGLAS

BJT	<i>Bipolar Junction Transistor</i>
CSV	<i>Comma Separated Values</i>
ENIAC	<i>Electronic Numerical Integrator and Computer</i>
FCCC	Fonte de corrente controlada por corrente
FCCT	Fonte de corrente controlada por tensão
FTCC	Fonte de tensão controlada por corrente
FTCT	Fonte de tensão controlada por tensão
GPS	<i>Global Positioning System</i>
GUI	<i>Graphical User Interface</i>
IDE	<i>Integrated Development Environment</i>
MOSFET	<i>Metal Oxide Semiconductor Field Effect Transistor</i>
NASA	<i>National Aeronautics and Space Administration</i>
PSIM	<i>PowerSim</i>
QUCS	<i>Quite Universal Circuit Simulator</i>
RAM	<i>Random Access Memory</i>
SFML	<i>Simple and Fast Multimedia Library</i>
SPICE	<i>Simulation Program with Integrated Circuit Emphasis</i>
SSD	<i>Solid State Drive</i>

LISTA DE SÍMBOLOS

n	Número de nós no circuito (não incluindo a referência)
m	Número de fontes de tensão independentes no circuito
A	Matriz de admitância estendida
x	Vetor de tensões nos nós e correntes nas fontes de tensão independentes
z	Vetor associado às fontes de correntes e fontes de tensão
G	Submatriz de A com dimensões $n \times n$
B	Submatriz de A com dimensões $n \times m$
C	Submatriz de A com dimensões $m \times n$
D	Submatriz de A com dimensões $m \times m$
v	Submatriz de x com dimensões $n \times 1$
j	Submatriz de x com dimensões $m \times 1$
i	Submatriz de z com dimensões $n \times 1$
e	Submatriz de z com dimensões $m \times 1$
R	Resistência do resistor
I	Corrente da fonte independente de corrente
V	Tensão da fonte independente de tensão
G_{FCCT}	Ganho da fonte de corrente controlada por tensão
A_{FTCT}	Ganho da fonte de tensão controlada por tensão
G_{FCCC}	Ganho da fonte de corrente controlada por corrente
A_{FTCC}	Ganho da fonte de tensão controlada por corrente
M	Matriz estendida da equação matricial $Ax = z$
M'	Matriz estendida após operações com linhas de M
x_i	i -ésimo elemento da matriz coluna x
L	Matriz triangular inferior correspondente à decomposição LU de A
U	Matriz triangular superior correspondente à decomposição LU de A
y_i	i -ésimo elemento da matriz coluna y
$v[n]$	Tensão na iteração presente
$v[n - 1]$	Tensão na iteração anterior
reltol	Tolerância relativa para o erro entre duas iterações consecutivas
vntol	Tolerância absoluta para as diferenças entre duas tensões consecutivas
g_{min}	Condutância mínima, aplicada em circuitos não lineares
I_D	Corrente do diodo

V_D	Tensão sobre o diodo
I_{SD}	Corrente de saturação do diodo
V_{TD}	Tensão térmica do diodo
V_{D0}	Tensão sobre o diodo em um dado ponto de operação
I_{D0}	Corrente no diodo em um dado ponto de operação
I_{Deq}	Valor da fonte de corrente do modelo linearizado do diodo
G_{Deq}	Valor da condutância do modelo linearizado do diodo
I_{es}	Corrente de saturação do emissor do BJT
I_{cs}	Corrente de saturação do coletor do BJT
V_{Te}	Tensão térmica da junção base-emissor do BJT
V_{Tc}	Tensão térmica da junção base-coletor do BJT
α_F	Ganho direto de corrente do BJT
α_R	Ganho reverso de corrente do BJT
g_{cc}	Oposto da condutância do coletor à base no modelo linearizado do BJT
g_{ee}	Oposto da condutância da base ao emissor no modelo linearizado do BJT
g_{ce}	Ganho da FTCT conectada entre o coletor e a base no modelo linearizado do BJT
g_{ec}	Ganho da FTCT conectada entre o emissor e a base no modelo linearizado do BJT
i_{c_eq}	Valor da fonte de corrente conectada entre o coletor e a base do no modelo linearizado do BJT
i_{e_eq}	Valor da fonte de corrente conectada entre o emissor e a base do no modelo linearizado do BJT
v_{gs}	Tensão entre a porta e a fonte do MOSFET
v_{ds}	Tensão entre o dreno e a fonte do MOSFET
v_{th}	Tensão limiar do MOSFET
I_{mos}	Corrente do dreno à fonte no MOSFET
K	Constante que representa as características construtivas do MOSFET
g_m	Ganho da FCCT do modelo linearizado do MOSFET
g_o	Condutância do modelo linearizado do MOSFET
V_a	Tensão de Efeito Early do MOSFET
h	Passo em tempo (<i>step</i>) usado no método de Euler
y_{n+1}	Aproximação da solução da equação diferencial no tempo t_{n+1}
y_n	Aproximação da solução da equação diferencial no tempo t_n
$f(t_{n+1}, y_{n+1})$	Derivada de y em t_{n+1}
$f(t_n, y_n)$	Derivada de y em t_n

I_C	Corrente do capacitor
C	Capacitância do capacitor
$v_c(t+h)$	Tensão do capacitor no tempo $t+h$
$v_c(t)$	Tensão do capacitor no tempo t
V_{Ceq}	Tensão da fonte de tensão no modelo do capacitor
R_{Ceq}	Resistência do resistor no modelo do capacitor
L	Indutância do indutor
$I_{L(n)}$	Corrente do indutor na iteração n
$V_{L(n)}$	Tensão do indutor na iteração n
I_{Leq}	Corrente da fonte de corrente no modelo do indutor
G_{Leq}	Condutância do resistor no modelo do indutor
Erros	Vetor das diferenças entre as tensões/correntes dos simuladores próprio e de referência
$V_{simul\ próprio}$	Vetor de pontos obtidos pelo simulador próprio
$V_{simul\ ref}$	Vetor de pontos obtidos pelo simulador de referência
$EA_{máximo}$	Erro absoluto máximo
$EA_{médio}$	Erro médio absoluto
V_{absmax}	Valor absoluto máximo de um conjunto de pontos de uma grandeza

SUMÁRIO

1	INTRODUÇÃO	17
1.1	Justificativa	20
1.2	Objetivos	22
1.2.1	Objetivo Geral	22
1.2.2	Objetivos Específicos	23
2	REFERENCIAL TEÓRICO	24
2.1	Descrição de um Circuito no SPICE	24
2.2	Tipos de Simulações	25
2.3	Análise Nodal Modificada	26
2.3.1	Análise para resistências e fontes independentes de tensão e corrente	26
2.3.2	Análise de Fontes Controladas	32
2.4	Métodos para Resolução de Sistemas de Equações Lineares	35
2.5	Análise de Circuitos Não Lineares	38
2.5.1	Método de Newton-Raphson	38
2.5.2	Modelo Linear para o Diodo	41
2.5.3	Modelo Linear para o Transistor Bipolar de Junção	43
2.5.4	Modelo Linear para o Transistor de Efeito de Campo Metal-Óxido-Semicondutor	48
2.6	Análise de Circuitos Variantes no Tempo	50
3	ETAPAS DE DESENVOLVIMENTO E METODOLOGIA	55
3.1	Etapas de Desenvolvimento	55
3.1.1	Linguagem de programação	55
3.1.2	Ferramentas auxiliares	56
3.1.3	Biblioteca de matrizes	57
3.1.4	Implementação do algoritmo para análises transientes	57
3.1.5	Interface gráfica para montagem de circuitos.....	60
3.1.6	Interface gráfica para exibição dos resultados.....	61
3.1.7	Exportação dos resultados para arquivos CSV	62
3.1.8	Armazenamento de circuitos em arquivos de texto.....	62
3.2	Metodologia	63
3.2.1	Classificação da pesquisa	63

3.2.2	Avaliação do desempenho do simulador	63
4	RESULTADOS E DISCUSSÃO	66
4.1	Interface Gráfica.....	66
4.2	Desempenho do Simulador	69
4.2.1	Precisão dos resultados	69
4.2.2	Usabilidade do simulador	85
5	CONCLUSÕES	88
	REFERÊNCIAS BIBLIOGRÁFICAS.....	90

1 INTRODUÇÃO

Desde a concepção dos primeiros computadores, simulações se tornaram ferramentas essenciais no desenvolvimento de novas tecnologias. Por proporcionarem uma forma pouco custosa e customizável de reproduzir comportamentos de uma variedade de processos, simulações são itens quase sempre presentes em projetos nas diversas áreas de engenharia.

O conceito de simulação é amplo e apresenta uma variedade de definições apresentadas por diferentes autores. Segundo o dicionário de língua portuguesa Michaelis (2021), a palavra “simulação” pode ser definida como a reprodução do funcionamento de um processo através do funcionamento de outro. Segundo Shannon (1998), uma simulação consiste no processo de modelar um sistema real e conduzir experimentos com o modelo visando compreender o comportamento do sistema e/ou avaliar estratégias para sua operação.

Simulações possuem aplicações diversas. Na indústria, por exemplo, são empregadas em setores como a otimização de processos (NORDGREN, 2002), segurança do trabalho e treinamento de funcionários. No contexto de economia, são amplamente utilizadas em modelos de crescimento e distribuição (SPINOLA, 2014). Mais recentemente, passaram a ser aplicadas na indústria de entretenimento, sendo incluídas em *video games*, filmes e análise de esportes. Modelos de simulação também são fundamentais em serviços como meteorologia e Sistema de Posicionamento Global (GPS), que são essenciais para os setores de transportes, agropecuária, segurança, dentre outros. No contexto acadêmico, simulações têm aplicações em diversos campos de estudo, de forma notável nas áreas de matemática, física e engenharia.

Em projetos de grande escala, em que ensaios e experimentos podem ter consequências significativas no aspecto econômico, nos recursos humanos e no tempo, o uso de simulações tem se tornado cada vez mais comum. Devido à crescente complexidade dos empreendimentos desenvolvidos por instituições de engenharia e tecnologia, a tarefa de prever e analisar todos os efeitos de alguma agregação ou alteração é extremamente laboriosa sem o uso de algum programa de computador, visto que o número de parâmetros variáveis ao longo de todo o projeto pode ser considerável. Nesse cenário, *softwares* de simulação se tornaram uma ferramenta imprescindível para o planejamento, projeto e controle de sistemas.

Simulações também possuem uso indispensável no estudo de sistemas que ainda não existem, não são devidamente compreendidos ou são impossíveis de reproduzir de forma controlada (ou de forma alguma), na prática. Isso pode ser constatado em áreas como física, química, biologia, entre várias outras. Como exemplo, pode-se citar a recente missão ao planeta Marte realizada pela agência espacial americana NASA, que está ocorrendo através do *rover* planetário *Perseverance*. Desde sua concepção, o projeto teve, em várias instâncias, auxílio de ambientes simulados por computador devido à dificuldade (ou até impossibilidade) de reproduzir com precisão as condições de Marte na Terra.

O desenvolvimento dos primeiros simuladores computacionais está aliado à concepção dos primeiros computadores eletrônicos. O ENIAC (*Electronic Numerical Integrator and Computer*), que é considerado o primeiro computador programável de grande escala (BURKS; BURKS, 1981), foi desenvolvido em 1943 pelos cientistas americanos John Eckert e John Mauchly, com o objetivo de auxiliar em missões militares. O aparato foi projetado com o intuito de realizar o cálculo de trajetórias balísticas para armamentos desenvolvidos no fim da Segunda Guerra Mundial, que até então era feito de forma manual por uma equipe de matemáticos (ROSEN, 1990). Em 1942, Mauchly escreveu em *The use of high speed vacuum tube devices for calculating* que, cálculos que demandariam de 15 a 30 minutos nos computadores analógicos mecânicos disponíveis até então, poderiam ser realizados em apenas 100 segundos com dispositivos eletrônicos (MAUCHLY, 1982). A considerável superioridade do desempenho dessa máquina em comparação com as outras de sua época estava associada ao uso de tubos de vácuo, que permitiam operações mais rápidas comparadas aos elementos eletromecânicos usados até então. O ENIAC iniciou a transição dos computadores eletromecânicos para versões baseadas em componentes eletrônicos, sendo que seu uso principal pode ser considerado uma das primeiras formas de simulação computacional (GOLDSMAN; NANCE; WILSON, 2009).

No contexto da engenharia elétrica, modelos de simulação são frequentemente utilizados no estudo de circuitos elétricos. No início da década de 1950, computadores eletromecânicos já eram utilizados para calcular as condições de equilíbrio de circuitos lineares (PEDERSON, 1984), permitindo, entre outras aplicações, a otimização de filtros elétricos. O posterior desenvolvimento dos computadores digitais foi acompanhando pelo aumento do uso de simuladores para fins de análises de circuitos elétricos.

Outro estímulo para a melhora em *softwares* de simulação foi o desenvolvimento dos semicondutores, que introduziram a possibilidade de haver componentes não lineares a circuitos elétricos cuja análise manual é muito dispendiosa e, na maioria das vezes, exige o uso de modelos simplificados para serem resolvidos de forma analítica. O transistor, por exemplo, que foi um dos primeiros dispositivos a ser criado com materiais dessa espécie, possui um comportamento particular que permite uma série de aplicações e, portanto, trouxe uma revolução na eletrônica do século XX. Apesar de ser um dos componentes mais utilizados atualmente e da grande quantidade de estudos a ele relacionados, a análise de circuitos com múltiplos transistores ainda é comumente feita através de simuladores, devido à dificuldade de modelar a característica não-linear desses elementos.

No cenário atual, a microeletrônica, em que se trabalha com circuitos de grande escala (podendo até incluir milhões de componentes) é outro campo em que a simulação de circuitos elétricos é muito demandada. O contínuo progresso no desempenho de peças que integram computadores como processadores, placas gráficas, unidades de estado sólido (SSD, do inglês *solid state drive*) e memórias de acesso aleatório (RAM, do inglês *random access memory*) vem acompanhado da necessidade de se utilizar componentes eletrônicos cada vez menores, podendo chegar à ordem de nanômetros. Consequentemente, exige-se o desenvolvimento de modelos bastante detalhados, que possam reproduzir o comportamento dos materiais a nível quântico.

O avanço da tecnologia também trouxe um barateamento de equipamentos eletrônicos, que facilitou o acesso a computadores pessoais para uma grande parcela da população. Nesse contexto, simulações podem ser utilizadas como complemento ao ensino. Para estudantes em áreas relacionadas à engenharia elétrica, simuladores atuam como uma ferramenta poderosa de suporte à aprendizagem, pois permitem visualização do comportamento de circuitos de uma forma relativamente próxima à realidade sem a necessidade de equipamentos de laboratório e uma montagem física de componentes. Também são úteis aos professores, que podem rapidamente construir ou modificar topologias como exemplo ou com o intuito de sanar dúvidas a respeito de um determinado conceito.

Com base na relevância desse tema, o projeto de graduação visou desenvolver um simulador de circuitos elétricos que possa simular circuitos com uma série de componentes básicos

abordados em engenharia elétrica. Este trabalho também apresenta um estudo do funcionamento de simuladores de circuitos elétricos já existentes, como parte da concepção de uma ferramenta de simulação focada em usabilidade e que possa ser aplicada no contexto acadêmico.

1.1 Justificativa

Como já discutido anteriormente, o uso de simuladores com o objetivo de analisar circuitos elétricos não é uma prática recente. Um dos primeiros programas a ser desenvolvido com esse intuito foi o SPICE, que foi anunciado em 1973 por Laurence Nagel e Donald Pederson (NAGEL; PEDERSON, 1973). O *software* foi escrito na linguagem de programação Fortran e possui licença de domínio público. Outras versões desse simulador surgiram nos anos seguintes, incluindo o SPICE2 e o SPICE3; sendo que o último foi programado em C. Por se tratar de um programa *open-source*, ou seja, seus direitos autorais permitem o estudo, modificação e redistribuição de seu código, o SPICE foi utilizado como base para uma série de outros programas sucessores, como o Pspice, LTSpice e NgSpice, entre outros.

Outro programa de simulação já consolidado no mercado é o PSIM (*PowerSim*), que foi disponibilizado pela primeira vez em 1994. Esta ferramenta foi desenvolvida especificamente para simulações envolvendo eletrônica de potência, embora também possua outras aplicações. O QUCS (*Quite Universal Circuit Simulator*) também é um simulador bastante conhecido. Trata-se de um *software* gratuito, lançado em 2003, que se tornou bastante popular no meio acadêmico pelo fato de apresentar uma interface mais simples e fácil de utilizar que simuladores como o PSpice.

Em sua seção de perguntas frequentes, os desenvolvedores do simulador QUCS, que não se baseia em SPICE, descrevem alguns motivos que favorecem a criação de uma ferramenta de simulação independente (QUCS TEAM, 2017). No texto, eles descrevem algumas restrições técnicas do SPICE3f.5 (versão mais recente do SPICE), além de apresentar algumas limitações impostas pela licença dessa versão. Outro aspecto comentado é a interface da aplicação, que pode ser pouco amigável para alguns usuários. Em resumo, o texto explica que a principal vantagem de desenvolver um simulador próprio é a implementação de recursos que atendem demandas específicas para um certo grupo de utilizadores.

A aplicação desenvolvida neste projeto é parte de um estudo aprofundado dos conceitos teóricos e práticos que envolvem a elaboração de um simulador, podendo contribuir como uma ferramenta didática para esse fim. Pode-se considerar que o seu principal diferencial em relação a outros simuladores seja sua interface simples e intuitiva, voltada especialmente para estudantes. Entretanto, o *software* final tem caráter educativo, não visando competir comercialmente com ferramentas já consolidadas no mercado.

Como forma de proporcionar um entendimento mais aprofundado sobre o assunto, o *software* foi concebido desde a geração de modelos a partir da disposição dos componentes do circuito até a resolução dos sistemas de equações diferenciais resultantes, sem o uso de quaisquer ferramentas de código adicionais que abstraíam os métodos matemáticos. Dessa forma, todos os algoritmos computacionais envolvidos – incluindo aqueles de operações com matrizes e resolução de sistemas lineares – foram implementados puramente em C++, sem o auxílio de bibliotecas. A única ferramenta empregada no projeto foi a biblioteca SFML, que permite a criação da interface gráfica da aplicação, assim como a interação do usuário com seus elementos através do teclado e *mouse*.

Um dos principais motivos que favoreceram a decisão de escrever a parte matemática do código sem o auxílio de bibliotecas foi a intenção de realizar um estudo detalhado de todas as etapas envolvidas no processo de simulação de um circuito elétrico. Outra vantagem foi a possibilidade de explorar e aplicar algumas técnicas de otimização para os algoritmos de operações com matrizes, o que não seria possível caso houvesse sido utilizada uma biblioteca com esse intuito. Além disso, o estudo de todo o mecanismo interno de funcionamento de um simulador pode auxiliar na solução de problemas que costumam surgir ao utilizar simuladores já estabelecidos no mercado, auxiliando tanto na compreensão de erros quanto na sua respectiva solução.

Outra vantagem de realizar a programação completa de um simulador foi o contexto educacional: o escopo do *software* desse projeto contemplou conteúdos de várias disciplinas ministradas ao longo do curso de Engenharia Elétrica, incluindo Cálculo, Álgebra Linear, Circuitos Elétricos, Eletrônica, Algoritmos Numéricos, além de tópicos abordados na pós-graduação, como o desenvolvimento de interfaces gráficas, discutido na disciplina de Técnicas Avançadas de Programação. A compreensão do funcionamento de um simulador de circuitos

elétricos poderia, portanto, atuar como motivação a alunos ingressantes e como ferramenta prática no estudo de conteúdos altamente teóricos.

Outra razão relevante para escolha do tema do projeto foi a limitação decorrente das restrições impostas por conta da pandemia do coronavírus. Muitas instituições de ensino estão operando de forma remota, tornando o uso dos laboratórios e outros ambientes de pesquisa difícil e/ou limitado. O desenvolvimento de um simulador foi uma tarefa que pôde ser feita majoritariamente através de um computador pessoal, possibilitando sua realização até mesmo sem acesso às instalações físicas da universidade.

1.2 Objetivos

1.2.1 Objetivo Geral

A primeira etapa no desenvolvimento de um simulador de circuitos elétricos foi a definição do seu escopo. Segundo Ngada (2014), alguns dos fatores esperados de uma boa ferramenta de simulação são:

- a) Interface que permita o desenho do circuito de forma confortável e intuitiva;
- b) Mensagens de erro de fácil interpretação pelo usuário;
- c) Execução robusta da simulação;
- d) Resultados da simulação podem ser exportados para outros programas para análise.

Tendo em vista estas demandas, o projeto visou desenvolver um simulador de circuitos elétricos que possuísse uma interface simples e intuitiva, sendo de fácil uso tanto para usuários experientes quanto iniciantes na área de engenharia elétrica. A aplicação deve simular topologias com componentes básicos de circuitos elétricos – incluindo fontes de tensão e corrente, resistores, capacitores e indutores – e alguns componentes não lineares, como diodos, transistores bipolares de junção (BJT, do inglês *bipolar junction transistor*) e transistores de efeito de campo metal-óxido-semicondutor (MOSFET, do inglês *metal oxide semiconductor field effect transistor*). O *software* final deve ser capaz de simular circuitos com os dispositivos citados com uma precisão razoável em comparação a simuladores já consolidados, de forma que seu tempo utilizado para gerar os dados de simulação também seja satisfatório.

1.2.2 Objetivos Específicos

Os objetivos específicos necessários para alcançar o objetivo geral do projeto foram:

- Criar um conjunto de rotinas que interprete a descrição de um circuito, gere modelos matemáticos dos componentes nele presentes e, por fim, resolva os sistemas de equações diferenciais resultantes;
- Desenvolver uma interface gráfica ou GUI (*Graphical User Interface*) que permita construir e modificar diagramas de circuitos elétricos;
- Produzir uma ferramenta que possa exibir os resultados da simulação na forma de gráficos de tensão (ou corrente) *versus* tempo. A análise dos resultados poderá ser auxiliada por funções básicas como *zoom* (aumento ou redução de escala) e *pan* (translação);
- Permitir ao usuário exportar os resultados da simulação para um arquivo no formato CSV (*Comma Separated Values*), que pode ser aberto por várias ferramentas de criação e edição de planilhas, como *Microsoft Excel* e *LibreOffice Calc*;
- Gerar uma versão do simulador que possa ser executada em computadores com sistemas operacionais baseados em *Linux* e *Windows* (multiplataforma);
- Avaliar o desempenho e usabilidade do programa final.

2 REFERENCIAL TEÓRICO

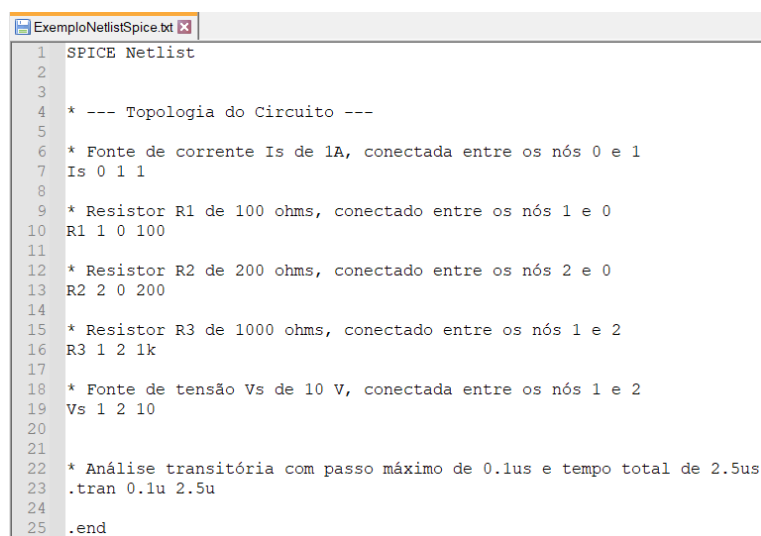
O conteúdo teórico abordado nesta seção será amplamente baseado no *software* SPICE, que, como já comentado anteriormente, é um dos programas de simulação mais consolidados e estudados no ramo da engenharia elétrica. Contudo, alguns processos são comuns à maioria das ferramentas de simulação, sendo, portanto discutidos de forma geral.

2.1 Descrição de um Circuito no SPICE

Em versões do SPICE que não possuem interface gráfica, a descrição de um circuito é dada através de um arquivo de texto conhecido como *netlist* (Figura 1), que contém todas as informações necessárias para a simulação incluindo: os nomes e tipos dos componentes; as ligações de seus terminais; seus valores (quando aplicável) e o tipo de análise a ser realizada assim como seus respectivos parâmetros.

Nesse arquivo, cada nó do circuito é associado a um número inteiro positivo, de forma que as ligações dos componentes são apresentadas com base nas conexões de cada terminal ao seu respectivo nó. Sendo assim a descrição do circuito é sempre a mesma independente de sua disposição visual em um diagrama, o que simplifica seu armazenamento em estruturas de dados e consequentemente facilita a sua análise através de um programa de computador.

Figura 1 – Exemplo de *netlist* do SPICE



```
1 SPICE Netlist
2
3
4 * --- Topologia do Circuito ---
5
6 * Fonte de corrente Is de 1A, conectada entre os nós 0 e 1
7 Is 0 1 1
8
9 * Resistor R1 de 100 ohms, conectado entre os nós 1 e 0
10 R1 1 0 100
11
12 * Resistor R2 de 200 ohms, conectado entre os nós 2 e 0
13 R2 2 0 200
14
15 * Resistor R3 de 1000 ohms, conectado entre os nós 1 e 2
16 R3 1 2 1k
17
18 * Fonte de tensão Vs de 10 V, conectada entre os nós 1 e 2
19 Vs 1 2 10
20
21
22 * Análise transitória com passo máximo de 0.1us e tempo total de 2.5us
23 .tran 0.1u 2.5u
24
25 .end
```

Fonte: Produção do próprio autor.

2.2 Tipos de Simulações

No ramo da simulação de circuitos, existem várias formas de realizar a sua resolução. A escolha de um método específico depende de vários fatores, como o tempo de simulação, o domínio no qual o circuito deve ser analisado (tempo ou frequência) e a presença de componentes que armazenam energia (capacitores e indutores).

O SPICE, por exemplo, possibilita que o usuário simule um circuito de 7 modos diferentes, a saber: *DC*; *AC*; *Transient*; *Pole-Zero*; *Small-Signal Distortion*; *Sensitivity* e *Noise* (NEWTON; PEDERSON; VINCENELLI, 1993). Desses, os três primeiros são os mais conhecidos e utilizados, sendo portanto abordados nesse projeto.

A análise *DC* de um circuito determina o seu ponto de operação considerando indutores como curto-circuitos e capacitores como circuitos abertos. Os modos *AC* e *Transient* sempre realizam uma varredura *DC* antes de suas respectivas etapas de simulação para determinar os modelos para pequenos sinais (*AC*) e as condições iniciais (*Transient*). Outra aplicação dessa análise é a geração de curvas de transferência, em que as variáveis de saída são armazenadas para cada valor de entrada especificado pelo usuário.

A análise *AC* do SPICE calcula os parâmetros de saída do circuito em função da frequência, tendo como base os modelos para pequenos sinais dos componentes. De forma resumida, este modo se baseia nas seguintes etapas: cálculo do ponto de operação do circuito através de uma varredura *DC*; determinação dos modelos linearizados para pequenos sinais nas condições encontradas e cálculo das variáveis desejadas para uma faixa de frequência definida pelo usuário.

A análise *Transient* (ou transiente, em português) tem a capacidade de simular circuitos complexos, podendo envolver dispositivos armazenadores de energia (capacitores e indutores) e não lineares (diodos, BJTs, etc.) no domínio do tempo. Por se tratar da forma mais comum de simulação, o projeto foi baseado nesse tipo de abordagem. Outro aspecto importante para essa escolha foi o fato de que a análise transiente possibilita simular circuitos operando em diversas condições, o que não é o caso para a avaliação *AC*, que se limita à obtenção de respostas para pequenos sinais.

2.3 Análise Nodal Modificada

A primeira etapa da análise do tipo *Transient* do SPICE consiste em encontrar as condições iniciais do circuito, a partir do cálculo das tensões em cada nó e das correntes em cada ramo. A obtenção de tais variáveis pode ser feita através das Leis de Kirchhoff, que enunciam que o somatório das correntes entrando ou saindo em um nó é igual a zero e que a soma das tensões em uma determinada malha é nula. A implementação dessas leis em simuladores geralmente se dá através de um método conhecido como análise nodal modificada.

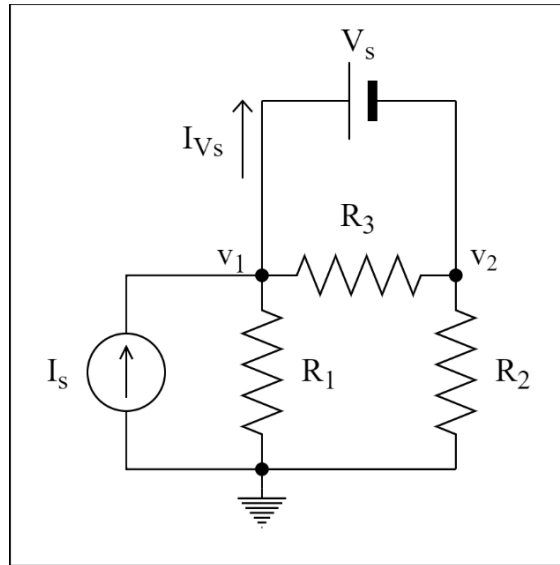
2.3.1 Análise para resistências e fontes independentes de tensão e corrente

De forma simplificada, o método da análise nodal modificada consiste das seguintes etapas (NILSSON; RIEDEL, 2011):

- Escolha arbitrária de um nó de referência, geralmente chamado de “Terra”, cuja tensão é definida como zero;
- Atribuição de números para os nós resultantes, de 1 a n , onde n é o número de nós que não são a referência. Cada par de nós com uma fonte de tensão independente entre eles é chamado pela literatura de “supernó”;
- Aplicação das Leis de Kirchhoff: formam-se equações para cada nó que não é a referência, de forma que a soma das correntes entrando e saindo de cada nó é igual zero. Para cada supernó, uma equação extra será formada relacionando as tensões desconhecidas de cada nó do par ao valor da fonte entre eles.
- Resolução do sistema de equações resultante. Para um circuito com n nós com tensões desconhecidas e m fontes de tensão independentes, o sistema linear possuirá $(n + m)$ incógnitas.

A solução será um vetor contendo as tensões em cada nó e as correntes em cada fonte de tensão. Para exemplificar esse processo, faz-se a análise do circuito representado na Figura 2.

Figura 2 – Circuito de exemplo para o método de análise nodal modificada



Fonte: Produção do próprio autor.

Primeiro, escolhe-se como referência o ponto ligado ao símbolo de terra, enquanto os outros nós são numerados por 1 e 2. Por conta da ligação da fonte de tensão V_s , 1 e 2 formam um supernó, gerando portanto uma equação a mais. Sabendo disso, obtém-se as equações (1), (2) e (3) através da aplicação das Leis de Kirchhoff:

$$\frac{v_1}{R_1} + \frac{v_1 - v_2}{R_3} + I_{V_s} - I_s = 0 \quad (1)$$

$$\frac{-v_1 + v_2}{R_3} + \frac{v_2}{R_2} - I_{V_s} = 0 \quad (2)$$

$$v_1 - v_2 - V_s = 0 \quad (3)$$

O circuito acima possui três nós ($n = 2$) e uma fonte de tensão independente ($m = 1$). Como já discutido, o sistema de equações resultante da aplicação do método deve possuir 3 ($n + m = 3$) variáveis de interesse, que são as tensões nos nós 1 e 2 (v_1 e v_2) e a corrente que passa por V_s (I_{V_s}).

Embora o equacionamento acima seja simples de ser obtido de forma analítica, seu armazenamento em programas de computador é mais fácil quando representado através de uma equação matricial. Desse modo, o sistema acima pode ser reescrito como mostra a equação (4):

$$\begin{bmatrix} \frac{1}{R_1} + \frac{1}{R_3} & -\frac{1}{R_3} & 1 \\ -\frac{1}{R_3} & \frac{1}{R_2} + \frac{1}{R_3} & -1 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ I_{V_S} \end{bmatrix} = \begin{bmatrix} I_S \\ 0 \\ V_S \end{bmatrix} \quad (4)$$

O processo descrito anteriormente pode ser sistematizado de forma geral para atender qualquer circuito composto de resistências e fontes de tensão e de corrente. O algoritmo da análise nodal modificada tem como objetivo preencher as matrizes A e z da equação (5), sendo que as grandezas desconhecidas são representadas pela matriz x (LITOVSKI; ZWOLINSKI, 1997). Nota-se que o formato da equação matricial em (5) é o mesmo que o encontrado na equação (4).

$$Ax = z \quad (5)$$

Na equação (5), A tem dimensões $(n + m) \times (n + m)$; x , $(n + m) \times 1$ e z também $(n + m) \times 1$.

Para facilitar o entendimento do método, pode-se ainda subdividir as matrizes em componentes menores, como mostram as equações (6), (7) e (8).

$$A = \begin{bmatrix} G & B \\ C & D \end{bmatrix} \quad (6)$$

$$x = \begin{bmatrix} v \\ j \end{bmatrix} \quad (7)$$

$$z = \begin{bmatrix} i \\ e \end{bmatrix} \quad (8)$$

Onde G tem dimensões $(n \times n)$; B , $(n \times m)$; C , $(m \times n)$ e D , $(m \times m)$. As matrizes v e i têm dimensão $(n \times 1)$, enquanto j e e têm dimensão $(m \times 1)$. O preenchimento será descrito a seguir.

Para a matriz G , cada elemento da diagonal principal é igual à soma de cada condutância (inverso da resistência) conectada ao nó correspondente à linha e coluna do elemento. Assim, o primeiro elemento da diagonal corresponde à soma de todas as condutâncias conectadas ao nó 1, enquanto o segundo corresponde à soma daquelas ligados ao nó 2, e assim por diante. Os elementos fora da diagonal são definidos como o oposto da soma das condutâncias conectadas

entre os nós descritos pela linha e coluna desse elemento. Então, um resistor conectado entre os nós 1 e 2 terá o oposto de sua condutância adicionada aos elementos (1, 2) e (2, 1) da matriz G . As matrizes B e C podem ser preenchidas com elementos iguais a 0, 1 ou -1 e se referem às ligações das fontes de tensão do circuito. Na matriz B , o elemento (u, k) é preenchido com 1, caso o terminal positivo da fonte u ($1 \leq u \leq m$) esteja conectado ao nó k ; e -1, caso contrário. Os elementos restantes são nulos. Para circuitos com apenas fontes independentes, C é igual à transposta de B . Ainda fazendo essa consideração, D é definida como uma matriz composta de zeros.

A matriz v corresponde ao vetor coluna com as tensões nos nós de 1 a n , enquanto j está relacionada às correntes sobre as fontes de tensão de 1 a m .

A matriz i corresponde à soma das correntes provenientes de fontes independentes de corrente que entram nos nós de 1 a n . Caso não haja fontes de corrente conectadas a um nó, seu respectivo elemento em i será igual a zero. Para a matriz e , por sua vez, cada linha corresponde aos valores das m fontes de tensão existentes no circuito.

Seguindo esta técnica para o circuito da Figura 2, encontra-se para a matriz A as equações matriciais 9, 10, 11 e 12:

$$G = \begin{bmatrix} \frac{1}{R_1} + \frac{1}{R_3} & -\frac{1}{R_3} \\ -\frac{1}{R_3} & \frac{1}{R_2} + \frac{1}{R_3} \end{bmatrix} \quad (9)$$

$$B = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \quad (10)$$

$$C = [1 \quad -1] \quad (11)$$

$$D = [0] \quad (12)$$

Para x , tem-se as equações (13) e (14):

$$v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \quad (13)$$

$$j = [I_{V_S}] \quad (14)$$

Para z , encontra-se as equações (15) e (16).

$$i = \begin{bmatrix} I_S \\ 0 \end{bmatrix} \quad (15)$$

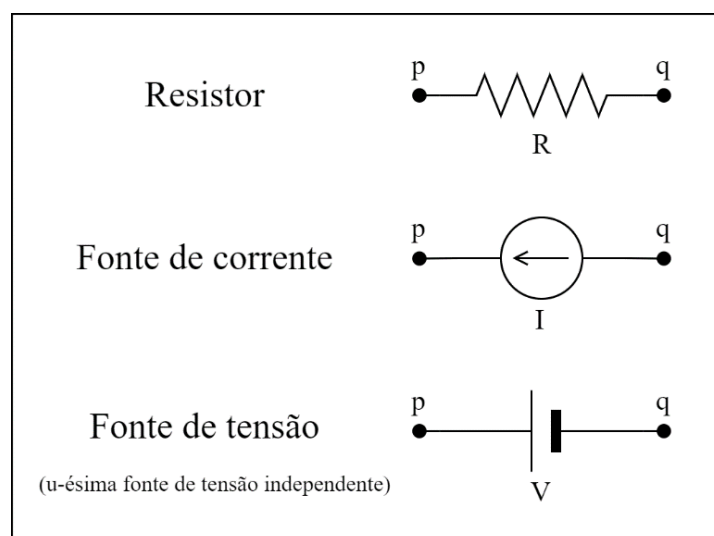
$$e = [V_S] \quad (16)$$

O concatenamento das matrizes obtidas pelas equações de (9) a (16) de acordo com (6) a (8), resulta no mesmo sistema obtido analiticamente, que é representado por (4).

Na implementação computacional do método de análise nodal modificada, as matrizes A , x e z são inicializadas com todos os seus elementos iguais a zero. Então, para cada componente do circuito em análise, os respectivos termos são somados aos elementos das submatrizes de interesse, resultando na equação (5). As parcelas a serem somadas podem ser descritas de forma resumida por meio de tabelas.

A Figura 3 mostra os parâmetros para um resistor e para fontes independentes de corrente e de tensão, a partir dos quais as tabelas a seguir foram definidas.

Figura 3 – Componentes lineares básicos



Fonte: Produção do próprio autor.

Um resistor de resistência R , ligado entre os nós p e q , resulta em termos que devem ser somados aos elementos associados às linhas e colunas p e q da submatriz G , como mostra a Tabela 1.

Tabela 1 – Termos que devem ser somados aos elementos da matriz G , para um resistor

Matriz G	Coluna p	Coluna q
Linha p	$1/R$	$-1/R$
Linha q	$-1/R$	$1/R$

Fonte: Produção do próprio autor.

Para uma fonte independente de corrente, apenas a submatriz i é alterada. Considerando que o seu valor é I e que está ligada entre os nós p e q , obtém-se a Tabela 2:

Tabela 2 – Termos que devem ser somados aos elementos da matriz i , para uma fonte independente de corrente

Matriz i	Coluna 1
Linha p	I
Linha q	$-I$

Fonte: Produção do próprio autor.

A u -ésima ($1 \leq u \leq m$) fonte independente de tensão de um circuito modifica as submatrizes B , C e e . Considerando que está conectada entre os nós p e q , são obtidas as Tabelas 3, 4 e 5.

Tabela 3 – Termos que devem ser somados aos elementos da matriz B , para uma fonte independente de tensão

Matriz B	Coluna u
Linha p	1
Linha q	-1

Fonte: Produção do próprio autor.

Tabela 4 – Termos que devem ser somados aos elementos da matriz C , para uma fonte independente de tensão

Matriz C	Coluna p	Coluna q
Linha u	1	-1

Fonte: Produção do próprio autor.

Tabela 5 – Termos que devem ser somados aos elementos da matriz e , para uma fonte independente de tensão

Matriz e	Coluna 1
Linha u	V

Fonte: Produção do próprio autor.

O método de análise nodal modificada também pode ser utilizado para encontrar pontos de operação de circuitos com componentes como capacitores, indutores e diodos. Isto é realizado através de uma série de procedimentos que transformam esses elementos em modelos lineares equivalentes, formados por fontes de corrente/tensão e resistências.

2.3.2 Análise de Fontes Controladas

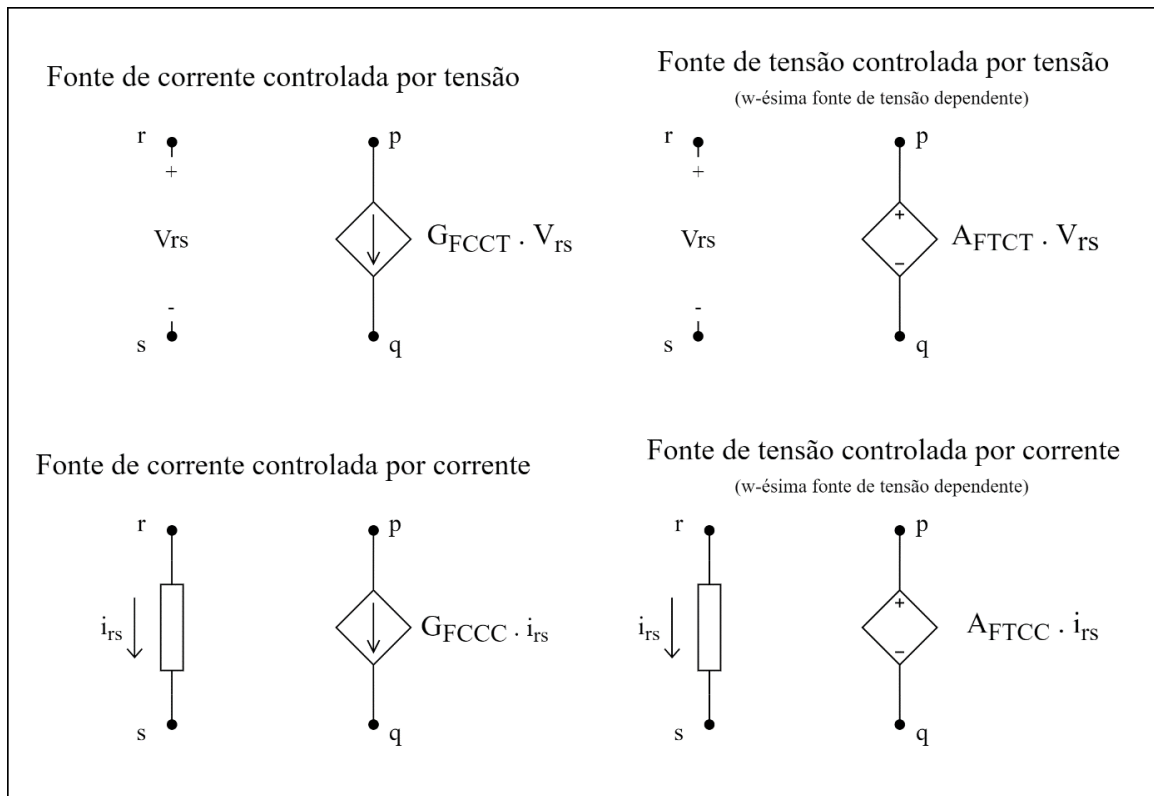
Embora o procedimento explicado até aqui seja suficiente para simular a maior parte dos componentes do simulador, alguns dispositivos não lineares, como BJTs e MOSFETs, incluem fontes controladas de tensão e/ou corrente em seus modelos. Portanto, esta seção inclui as tabelas necessárias para a consideração desses tipos de fonte na equação matricial do método da análise nodal modificada. Os valores acoplados a cada tabela foram baseados nos trabalhos de Queiroz (1995) e Jahn *et al.* (2007).

Os tipos de fontes dependentes são:

- Fonte de corrente controlada por tensão (FCCT);
- Fonte de tensão controlada por tensão (FTCT);
- Fonte de corrente controlada por corrente (FCCC);
- Fonte de tensão controlada por corrente (FTCC).

A Figura 4 mostra as conexões de cada tipo de fonte dependente.

Figura 4 – Fontes dependentes de tensão



Fonte: Produção do próprio autor.

Com o intuito de proporcionar uma representação mais compacta, as relações serão representadas com base nas matrizes principais (A e z).

Para uma fonte de corrente com ganho G_{FCCT} , ligada entre os nós p e q , controlada pela tensão entre os nós r e s , tem-se a formulação mais simples dentre os quatro tipos, como mostra a Tabela 6.

Tabela 6 – Termos que devem ser somados aos elementos da matriz A , para uma FCCT

Matriz A	Coluna r	Coluna s
Linha p	G_{FCCT}	$-G_{FCCT}$
Linha q	$-G_{FCCT}$	G_{FCCT}

Fonte: Produção do próprio autor.

Uma fonte de tensão com ganho A_{FTCT} , conectada entre os nós p e q , controladas pela tensão entre os nós r e s , resulta nos termos da Tabela 7. É importante observar que fontes de tensão controladas introduzem uma nova linha e coluna à matriz A , além de uma nova linha às matrizes

x e z , sendo estas geralmente acrescentadas após as $(n + m)$ linhas/colunas das matrizes obtidas para os componentes lineares básicos. Nas Tabelas 7 e 9, w se refere ao indicador da fonte de tensão dependente, que acarretou o acréscimo da $(n+m+w)$ -ésima linha à matriz A .

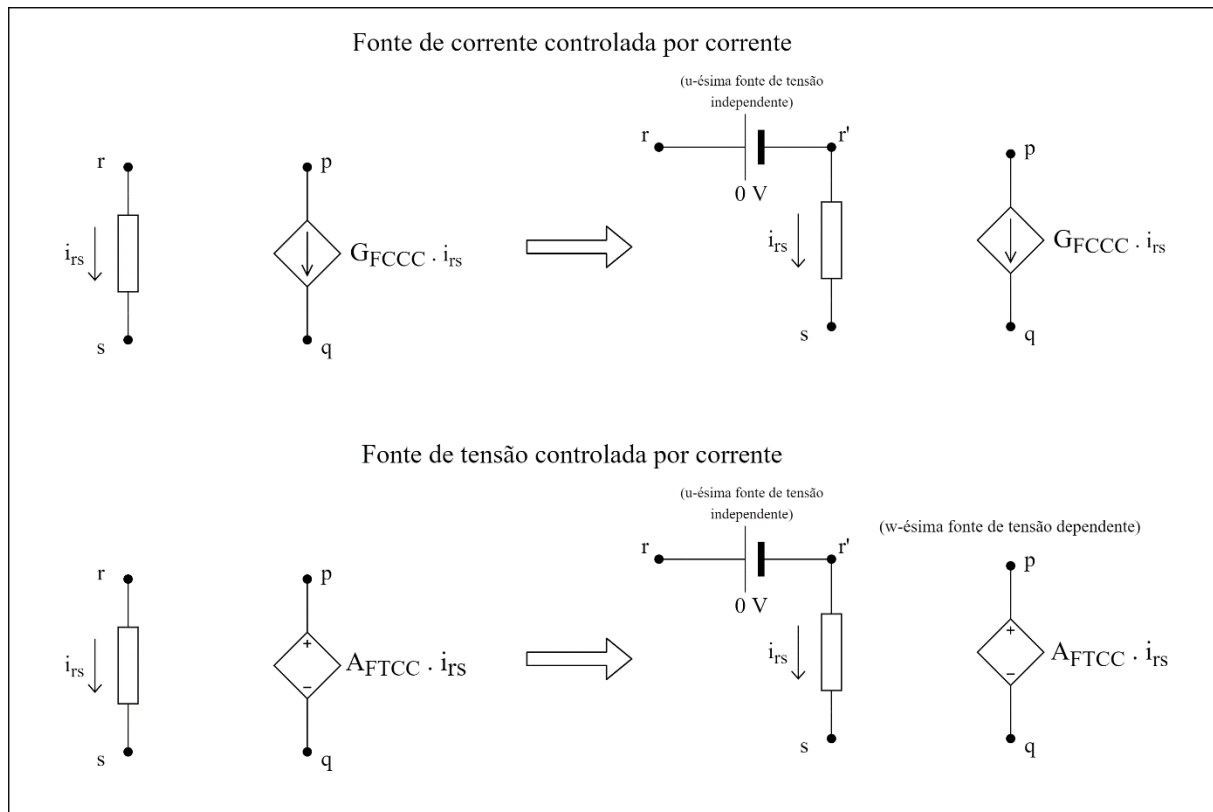
Tabela 7 – Termos que devem ser somados aos elementos da matriz A , para uma FTCT

Matriz A	Coluna p	Coluna q	Coluna r	Coluna s	Coluna $(n + m + w)$
Linha p	0	0	0	0	1
Linha q	0	0	0	0	-1
Linha $(n + m + w)$	1	-1	$-A_{FTCT}$	A_{FTCT}	0

Fonte: Produção do próprio autor.

Para facilitar a análise de fontes controladas por corrente, adiciona-se entre os nós do ramo controlador (r e s) uma fonte de tensão independente de 0 V, conseqüentemente também acrescentando um nó (r') ao circuito. Isso tem como objetivo facilitar a obtenção da corrente pelo ramo independentemente do(s) elemento(s) conectado(s) entre os dois nós. A Figura 5 ilustra esse procedimento.

Figura 5 – Procedimento para fontes controladas por corrente



Fonte: Produção do próprio autor.

Assim, uma fonte de corrente com ganho G_{FCCC} , ligada entre os nós p e q , controlada pela corrente que passa pela u -ésima ($1 \leq u \leq m$) fonte de tensão independente, resulta nas modificações descritas pela Tabela 8.

Tabela 8 – Termos que devem ser somados aos elementos da matriz A, para uma FCCC

Matriz A	Coluna p	Coluna q	Coluna r	Coluna r'	Coluna $(n + u)$
Linha p	0	0	0	0	G_{FCCC}
Linha q	0	0	0	0	$-G_{FCCC}$
Linha r	0	0	0	0	1
Linha r'	0	0	0	0	-1
Linha $(n + u)$	0	0	1	-1	0

Fonte: Produção do próprio autor.

Para uma fonte de tensão com ganho A_{FTCC} , conectada entre os nós p e q , controlada pela corrente que atravessa a u -ésima ($1 \leq u \leq m$) fonte de tensão independente do circuito, encontra-se a Tabela 9.

Tabela 9 – Termos que devem ser somados aos elementos da matriz A, para uma FTCC

Matriz A	Coluna p	Coluna q	Coluna r	Coluna r'	Coluna $(n + u)$	Coluna $(n + m + w)$
Linha p	0	0	0	0	0	1
Linha q	0	0	0	0	0	-1
Linha r	0	0	0	0	1	0
Linha r'	0	0	0	0	-1	0
Linha $(n + u)$	0	0	1	-1	0	0
Linha $(n + m + w)$	1	-1	0	0	$-A_{FTCC}$	0

Fonte: Produção do próprio autor.

2.4 Métodos para Resolução de Sistemas de Equações Lineares

Existem vários algoritmos que podem ser utilizados para encontrar a solução dos sistemas de equação lineares resultantes da aplicação do método de análise nodal modificada. Uma das técnicas mais conhecidas para realizar essa tarefa é o método da eliminação de Gauss (também chamado de escalonamento), que consiste na simplificação da matriz estendida do sistema

através de operações sucessivas com suas linhas (LAY; LAY; MCDONALD, 2015). A matriz estendida (M) da equação matricial em (5) é definida como a concatenação de A e z (mostrada em (17)), enquanto o resultado do escalonamento está representado em (18).

$$M = [A \mid z] \quad (17)$$

$$M' = \begin{bmatrix} a_{11} & * & * & \dots & * & b_1 \\ 0 & a_{22} & * & \dots & * & b_2 \\ \vdots & \vdots & \vdots & \dots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & a_{n+m,n+m} & b_{n+m} \end{bmatrix} \quad (18)$$

Para que o sistema original tenha solução exata, a matriz quadrada formada pelas colunas (exceto a última) de M' deve ser triangular superior, ou seja, $(a_{11}, a_{22}, \dots, a_{(n+m),(n+m)} \neq 0)$ deve ser verdadeira.

Caso as condições anteriormente citadas sejam atendidas, é possível aplicar o algoritmo de substituições retroativas para obter a solução do sistema. A equação (19) descreve como obter as soluções x_i para $i = n+m, n+m-1, \dots, 1$.

$$x_i = \frac{b_i - \sum_{j=i+1}^{n+m} (a_{ij} \cdot x_j)}{a_{ii}} \quad (19)$$

Em simulações no domínio do tempo, em que soluções de sistemas lineares devem ser computadas para cada passo discreto realizado, é comum que a matriz A resultante da aplicação do método de análise nodal modificada permaneça constante durante toda ou grande parte do período desejado para a análise, especialmente para circuitos com componentes exclusivamente lineares. Nesse contexto, o método de eliminação de Gauss torna-se ineficiente, pois sempre simplifica a matriz A para resolver o sistema.

Para melhorar o desempenho, simuladores à base do SPICE usam algoritmos de decomposição LU para realizar a solução de sistemas lineares. Essa técnica consiste em fatorar a matriz A em duas matrizes: uma triangular inferior L e outra triangular superior U , tal que a equação (20) seja verdadeira.

$$A = LU = \begin{bmatrix} l_{11} & 0 & \dots & 0 \\ l_{21} & l_{22} & \dots & 0 \\ \vdots & \dots & \ddots & \vdots \\ l_{m+n,1} & l_{m+n,2} & \dots & l_{m+n,m+n} \end{bmatrix} \cdot \begin{bmatrix} u_{11} & u_{12} & \dots & u_{1,m+n} \\ 0 & u_{22} & \dots & u_{2,m+n} \\ \vdots & \dots & \ddots & \vdots \\ 0 & 0 & \dots & u_{m+n,m+n} \end{bmatrix} \quad (20)$$

Dentre os diversos métodos que podem ser aplicados para realizar a decomposição, escolheu-se o algoritmo de Crout com pivotação, que tem a vantagem de reduzir erros de arredondamento e truncamento em decorrência de operações com números de ponto flutuante, tornando o método estável (PRESS; TEUKOLSKY; VETTERLING; FLANNERY, 2007).

Após a decomposição, pode-se reescrever a equação matricial em (5) como mostra a equação (21).

$$LUx = z \quad (21)$$

O procedimento para obter a solução de (21) é dividido em duas etapas. Primeiro, deve-se resolver a equação matricial intermediária $Ly = z$, que, devido à propriedade de L ser triangular inferior, pode ser resolvida através de um algoritmo similar ao de substituições retroativas, conhecido como substituições sucessivas. A equação (22) descreve este método, sendo y_i e z_i os i -ésimos elementos das matrizes coluna y e z , respectivamente. Os elementos nas linhas i e j de L são representados por l_{ij} .

$$y_i = \frac{z_i - \sum_{j=1}^{i-1} (l_{ij} \cdot y_j)}{l_{ii}} \quad (22)$$

Sabendo y , pode-se realizar a segunda etapa do procedimento, que consiste em resolver a equação $Ux = y$. Devido ao fato de U ser triangular superior, pode-se usar o algoritmo de substituições retroativas, já descrito na equação (19).

Nota-se que os fatores L e U são sempre os mesmos em sistemas em que matriz A permanece constante. Para essa condição, a aplicação dos algoritmos de substituições sucessivas e retroativas exige menos esforço computacional que a eliminação de Gauss (BLUME, 1986). Também é possível demonstrar que, mesmo que ocorra variação de A e a decomposição LU tenha que ser realizada em cada passo, o procedimento como um todo possuirá a mesma

complexidade computacional que a eliminação de Gauss (BLUME, 1986). De forma geral, o método de decomposição LU é vantajoso no melhor caso e similar no pior caso à eliminação de Gauss, justificando sua escolha em simuladores.

2.5 Análise de Circuitos Não Lineares

Os métodos abordados anteriormente baseiam-se na premissa de que todos os componentes do circuito analisado são lineares, não havendo a possibilidade de simular componentes não lineares diretamente. Contudo, é possível encontrar modelos lineares para esses elementos, de forma que o método de análise nodal modificada possa ser utilizado nessas condições.

2.5.1 Método de Newton-Raphson

Um dos requisitos para a obtenção do modelo linear para um componente não linear é o cálculo do seu ponto de operação, considerando o circuito em que ele está contido. Essa tarefa pode ser realizada através do método de Newton-Raphson, que consiste em uma técnica usada para obter aproximações das raízes de funções, dado que suas derivadas são conhecidas (PILLAGE; ROHRER; VISWESWARIAH, 1994).

De forma simplificada, o algoritmo do método de Newton-Raphson se baseia nas seguintes etapas:

- a) Escolha de um ponto de operação inicial qualquer para cada componente não-linear do circuito;
- b) Criação dos modelos lineares;
- c) Aplicação do método de análise nodal modificada e posterior resolução da equação matricial $Ax = z$ resultante;
- d) Verificação de convergência: esta etapa consiste na análise das tensões obtidas para cada componente em relação à última iteração. Caso a diferença percentual entre todas as tensões atuais e anteriores sejam menores que um limiar pré-estabelecido, considera-se que a solução é satisfatória e o procedimento é terminado; caso contrário, o ponto de operação deve ser recalculado, sendo necessário retornar à etapa b. É importante

observar que essa condição deve ser atendida para todos os componentes não lineares, havendo repetição do processo até que se alcance modelos aceitáveis para todos eles.

Existem diversos parâmetros que podem ser utilizados para determinar se a solução obtida para a equação matricial resultante da análise nodal modificada é suficientemente precisa para o término do ciclo de Newton-Raphson. Os mais conhecidos são chamados de *reltol*, *vntol* e *abstol* (KUNDERT; CLIFFORD, 1993).

O parâmetro *reltol* é definido como uma tolerância para o erro relativo entre duas iterações consecutivas. Por exemplo, caso o valor esperado para uma variável seja 1, e o *reltol* seja igual a 1%, é necessário que a diferença entre os valores das iterações seja igual a $1 \cdot 1\% = 0,01$ para que a solução seja aceita. À medida que o valor esperado para uma variável se aproxima de zero, as diferenças necessárias tornam-se cada vez menores, dificultando a chegada à convergência. Para isso, são utilizados os parâmetros *vntol* e *abstol*, que quantificam as tolerâncias absolutas para as diferenças entre os valores de tensão e corrente, respectivamente.

Com o objetivo de simplificar a implementação do simulador, optou-se por checar apenas a convergência das tensões dos componentes, utilizando um critério similar ao aplicado pelo SPICE, mostrado na equação (23). Esse método de verificação é o mesmo aplicado pelo simulador *Falstad Circuit Simulator* (FALSTAD, 2021).

$$|v[n] - v[n - 1]| < reltol \cdot \max(|v[n]|, |v[n - 1]|) + vntol \quad (23)$$

Onde $v[n]$ e $v[n - 1]$ são as tensões nas iterações presente e anterior, respectivamente.

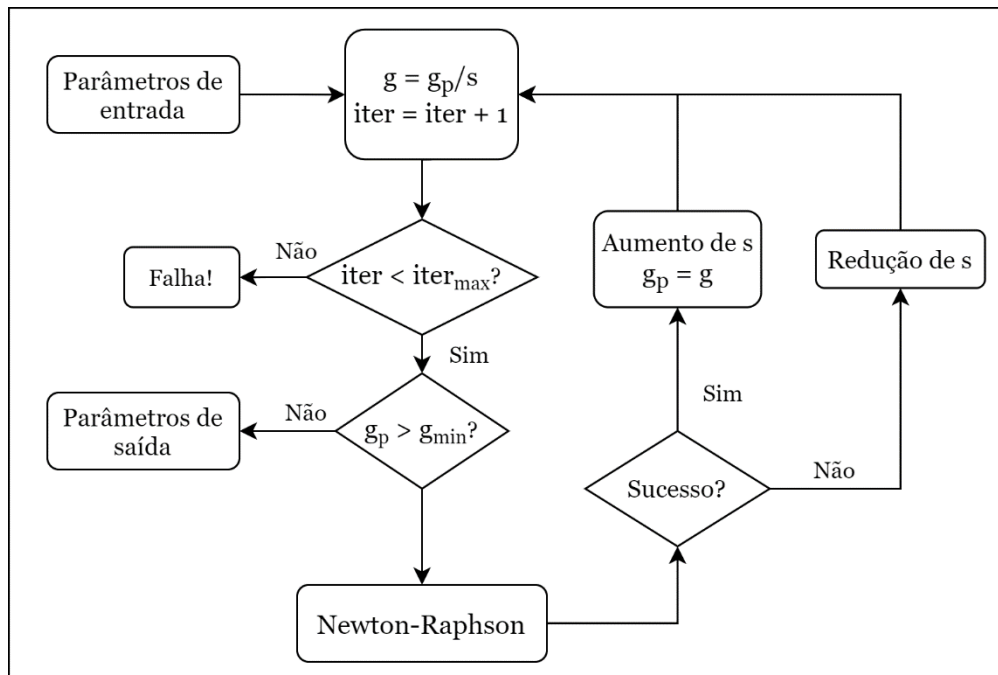
Em alguns casos, o algoritmo explicado anteriormente não consegue produzir soluções estáveis, exigindo muitas iterações computacionais para convergir. Portanto, faz-se necessário haver um número máximo de iterações, após as quais o processo é terminado independente do alcance ou não da convergência. Esse problema também pode ser amenizado com o uso de condições de convergência mais brandas ou através de um algoritmo auxiliar para a convergência.

Uma solução utilizada pelo simulador SPICE consiste no acoplamento de uma pequena condutância entre os terminais de dispositivos não lineares (KUNDERT; CLIFFORD, 1993).

Esse procedimento facilita a convergência dos valores em circuitos não lineares, com o custo de reduzir a precisão dos resultados, já que a análise é realizada em um circuito diferente do original.

Para minimizar o efeito do uso dessa técnica sobre a precisão, o SPICE usa uma condutância mínima (g_{min}) que tem seu valor reduzido a cada término bem-sucedido do algoritmo de Newton-Raphson. Assim, o resultado final pode ser atingido gradualmente após várias aplicações do ciclo, quando a condutância for pequena a ponto de sua influência ser desprezível. Tuma e Buermen (2009) descreveram um algoritmo detalhado para esse procedimento, que pode ser visto no diagrama da Figura 6.

Figura 6 – Algoritmo de redução gradual da condutância auxiliar



Fonte: Tuma e Buermen (2009).

Nota: Adaptado pelo autor.

Na Figura 6, g é a condutância atual sendo testada, enquanto g_p se refere à última condutância mínima que obteve sucesso na execução do método de Newton-Raphson. s é um valor que pode ser aumentado ou diminuído, dependendo da necessidade de redução ou aumento de g . $iter$ e $iter_{max}$ são os números de iterações parcial e máximo do algoritmo.

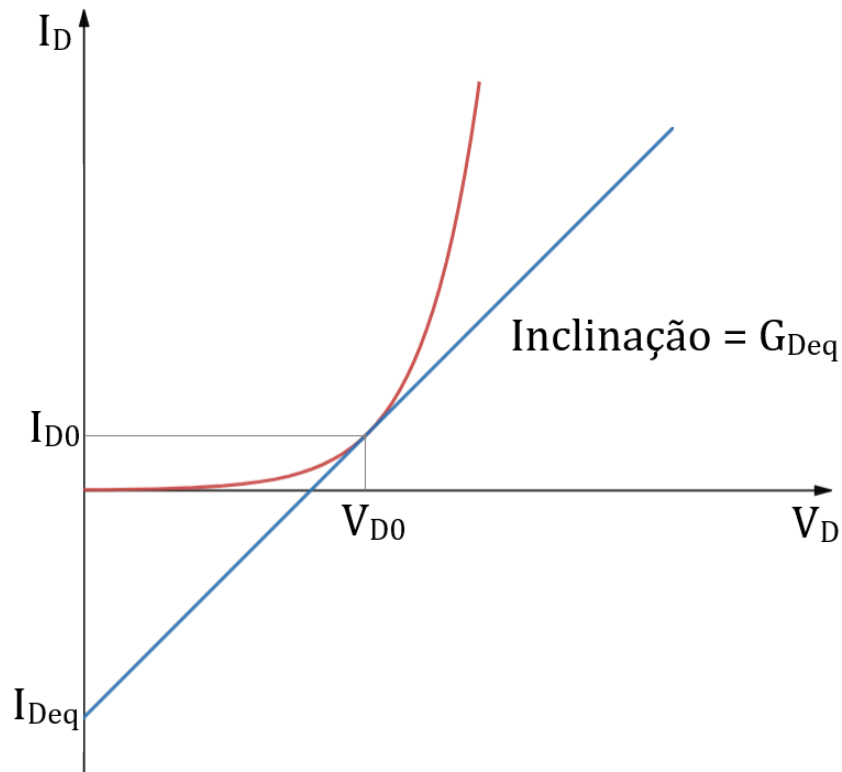
2.5.2 Modelo Linear para o Diodo

A relação de corrente (I_D) em função da tensão (V_D) de um diodo é dada pela equação (24), sendo V_{TD} e I_{SD} constantes.

$$I_D = I_{SD} \cdot \left(e^{\frac{V_D}{V_{TD}}} - 1 \right) \quad (24)$$

No Gráfico 1, a curva IxV está mostrada em vermelho, enquanto a reta em azul aproxima linearmente o comportamento do diodo para o ponto de operação (V_{D0} , I_{D0}).

Gráfico 1 – Curva IxV do diodo e reta que a aproxima no ponto (V_{D0} , I_{D0})



Fonte: Produção do próprio autor.

Para obter um modelo linear desse elemento, pode-se aproximar a equação (24) como os dois primeiros termos da série de Taylor (BLUME, 1986), cuja equação está representada em (25).

$$f(x) = f(a) + f'(a) \cdot (x - a) \quad (25)$$

Em (25), $f(x)$ corresponde a uma função de x qualquer, enquanto $f'(a)$ representa a sua derivada em relação a x na abscissa a . Para encontrar a aproximação é necessário calcular $f'(a)$, que nesse caso corresponde a condutância do diodo para a tensão V_{D0} . Seu cálculo é mostrado em (26):

$$G_{Deq} = \left[\frac{\partial I_D}{\partial V_D} \right]_{V_D=V_{D0}} = \frac{I_{SD}}{V_{TD}} e^{\frac{V_{D0}}{V_{TD}}} \quad (26)$$

Com o resultado de (26), pode-se obter a aproximação por série de Taylor de (25) para a tensão V_{D0} , que está representada na equação (27).

$$I_D = I_{SD} \cdot \left(e^{\frac{V_{D0}}{V_{TD}}} - 1 \right) + G_{Deq} \cdot (V_D - V_{D0}) \quad (27)$$

Nota-se que o primeiro termo do lado direito de (27) corresponde à corrente para V_{D0} , denominada I_{D0} . Reescrevendo, encontra-se a equação (28).

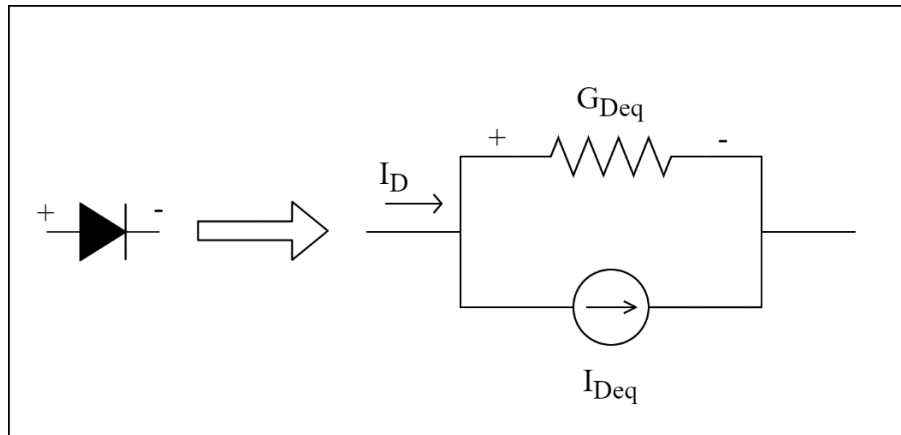
$$I_D = I_{D0} + G_{Deq} \cdot V_D - G_{Deq} \cdot V_{D0} \quad (28)$$

Chamando a parcela $(I_{D0} - G_{Deq} \cdot V_{D0})$ de I_{Deq} , obtém-se a equação (29), que está representada pela reta azul no Gráfico 1.

$$I_D = I_{Deq} + G_{Deq} \cdot V_D \quad (29)$$

Em termos de circuitos elétricos, (29) pode ser representada por uma associação em paralelo de uma fonte de corrente de valor I_{Deq} com uma condutância G_{Deq} (MCNEILL, 1986), como mostra a Figura 7:

Figura 7 – Circuito linear equivalente do diodo

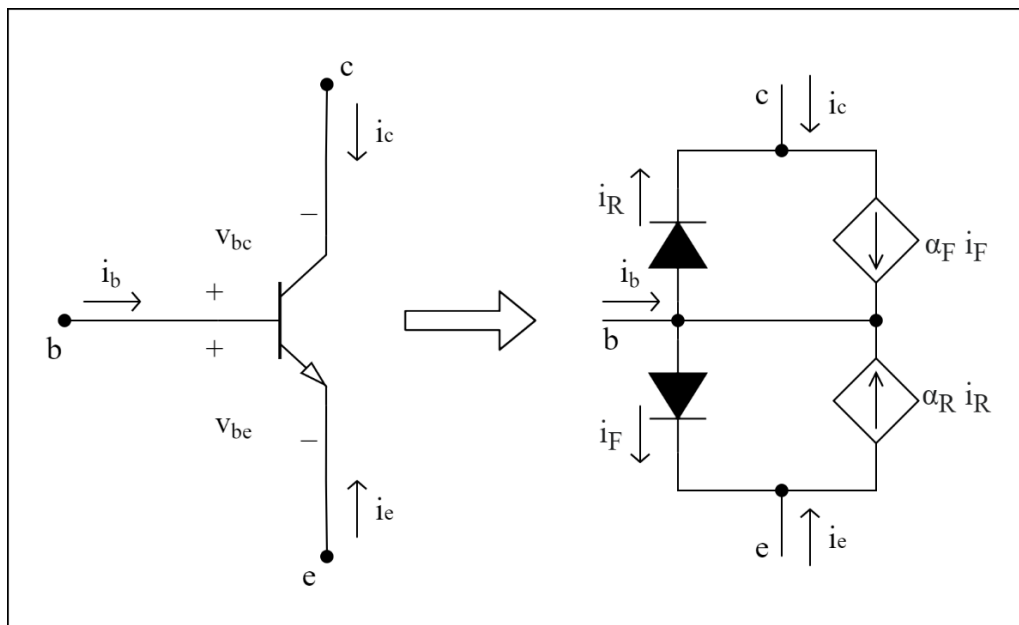


Fonte: Produção do próprio autor.

2.5.3 Modelo Linear para o Transistor Bipolar de Junção

O modelo matemático utilizado para simular transistores bipolares é conhecido como modelo de Ebers-Moll, que foi introduzido em 1954 por Jewell James Ebers e John L. Moll. A Figura 8 representa o diagrama para um BJT do tipo NPN.

Figura 8 – Modelo de Ebers-Moll para o BJT do tipo NPN



Fonte: Produção do próprio autor.

As equações que definem o comportamento de um transistor do tipo NPN são enunciadas por (30) e (31) (PILLAGE; ROHRER; VISWESWARIAH, 1994).

$$i_c = \alpha_F \cdot I_{es} \left(e^{\frac{v_{be}}{v_{Te}}} - 1 \right) - I_{cs} \left(e^{\frac{v_{bc}}{v_{Tc}}} - 1 \right) \quad (30)$$

$$i_e = -I_{es} \left(e^{\frac{v_{be}}{v_{Te}}} - 1 \right) + \alpha_R \cdot I_{cs} \left(e^{\frac{v_{bc}}{v_{Tc}}} - 1 \right) \quad (31)$$

Onde I_{es} e I_{cs} são as correntes de saturação do emissor e do coletor; v_{Te} e v_{Tc} são as tensões térmicas do transistor; e α_F e α_R são os ganhos diretos e reversos de corrente.

Os parâmetros do modelo podem ser obtidos através das equações (32), (33), (34) e (35).

$$\frac{\partial i_c}{\partial v_{bc}} = -\frac{I_{cs}}{v_{Tc}} \cdot e^{\frac{v_{bc}}{v_{Tc}}} = g_{cc} \quad (32)$$

$$\frac{\partial i_c}{\partial v_{be}} = \alpha_F \frac{I_{es}}{v_{Te}} \cdot e^{\frac{v_{be}}{v_{Te}}} = g_{ce} \quad (33)$$

$$\frac{\partial i_e}{\partial v_{bc}} = \alpha_R \frac{I_{cs}}{v_{Tc}} \cdot e^{\frac{v_{bc}}{v_{Tc}}} = g_{ec} \quad (34)$$

$$\frac{\partial i_e}{\partial v_{be}} = -\frac{I_{es}}{v_{Te}} \cdot e^{\frac{v_{be}}{v_{Te}}} = g_{ee} \quad (35)$$

É possível mostrar que a formulação do modelo acima resulta nas Tabelas 10 e 11, que representam o preenchimento das matrizes principais (A e z) do método de análise nodal modificada para um BJT do tipo NPN (PILLAGE; ROHRER; VISWESWARIAH, 1994). Considera-se que emissor, coletor e base estão ligados nos nós e , c e b , respectivamente.

Tabela 10 – Termos que devem ser somados aos elementos da matriz A , para um BJT do tipo NPN

Matriz A	Coluna e	Coluna c	Coluna b
Linha e	$-g_{ee}$	$-g_{ec}$	$g_{ee} + g_{ec}$
Linha c	$-g_{ce}$	$-g_{cc}$	$g_{ce} + g_{cc}$
Linha b	$g_{ee} + g_{ce}$	$g_{ec} + g_{cc}$	$-g_{ee} - g_{ec} - g_{ce} - g_{cc}$

Fonte: Produção do próprio autor.

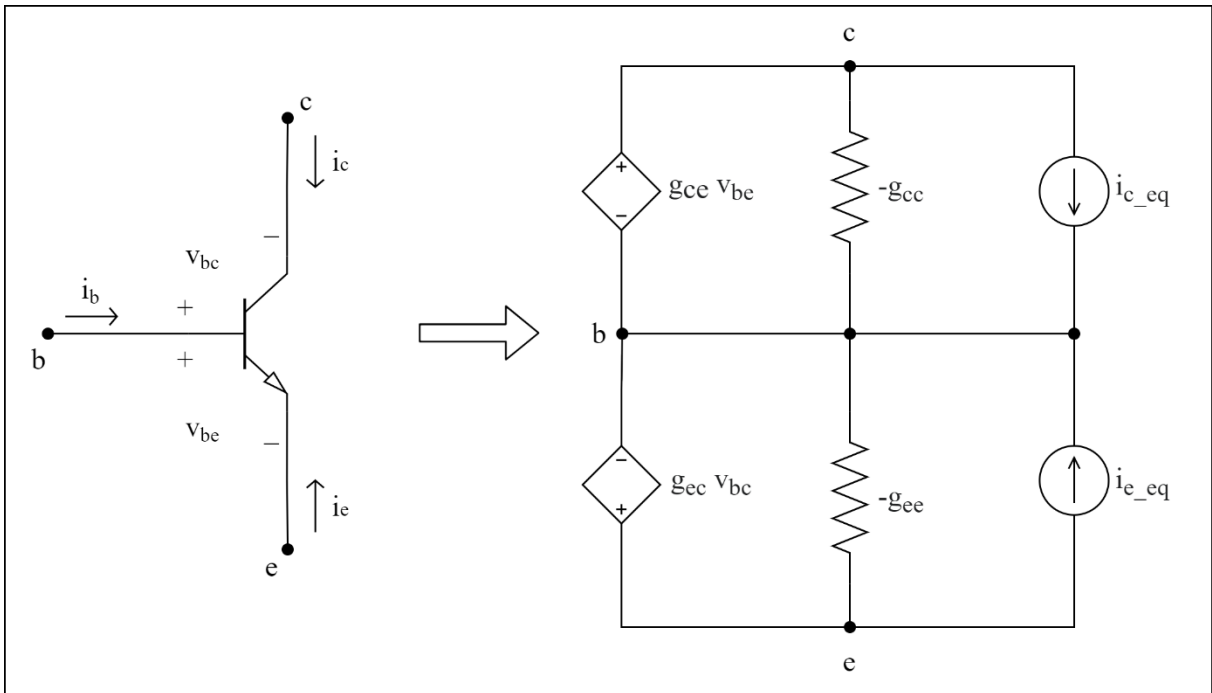
Tabela 11 – Termos que devem ser somados aos elementos da matriz z , para um BJT do tipo NPN

Matriz z	Coluna 1
Linha e	$-i_e$
Linha c	$-i_c$
Linha b	$i_e + i_c$

Fonte: Produção do próprio autor.

Embora as duas tabelas anteriores apresentem os resultados após a aplicação dos algoritmos já discutidos de forma direta, elas também podem ser obtidas a partir da análise do modelo da Figura 9, que é composto por fontes de tensão controladas por tensão, condutâncias e fontes de corrente independentes. Dessa forma, permite-se que transistores bipolares de junção sejam modelados como uma associação dos componentes básicos já tratados na Seção 2.3, facilitando sua integração às rotinas já implementadas previamente.

Figura 9 – Modelo para o BJT NPN baseado em componentes lineares



Fonte: Produção do próprio autor.

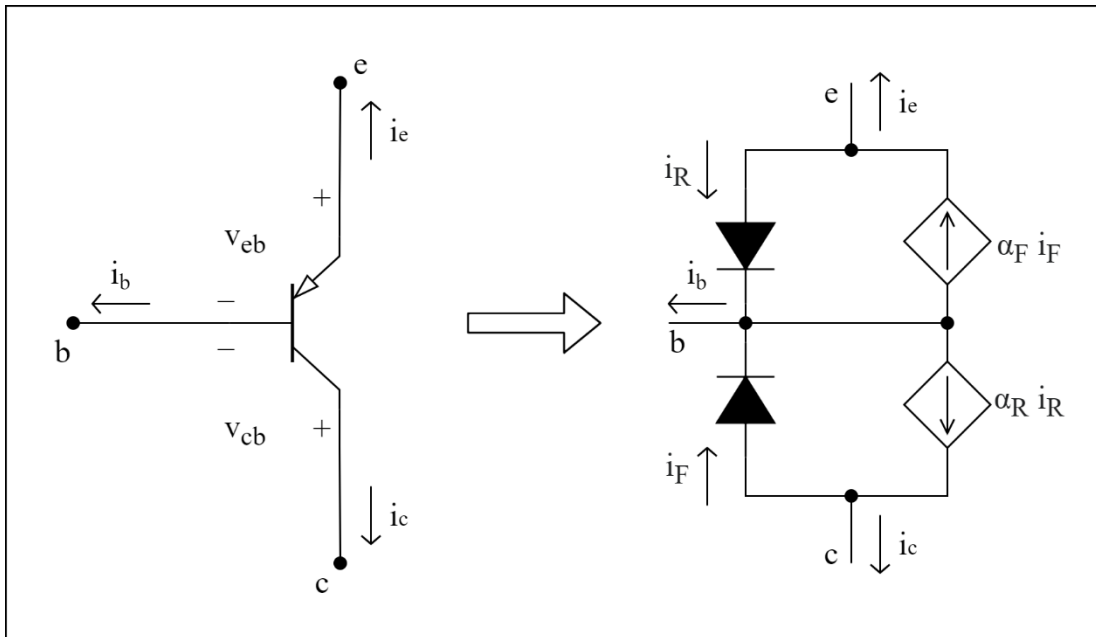
As equações (36) e (37) descrevem as correntes das fontes de corrente independentes i_{c_eq} e i_{e_eq} presentes na Figura 9.

$$i_{c_eq} = i_c - (g_{ce} \cdot v_{be} + g_{cc} \cdot v_{bc}) \quad (36)$$

$$i_{e_eq} = i_e - (g_{ee} \cdot v_{be} + g_{ec} \cdot v_{bc}) \quad (37)$$

A análise de transistores bipolares PNP é semelhante à realizada para aqueles do tipo NPN, como mostra a Figura 10.

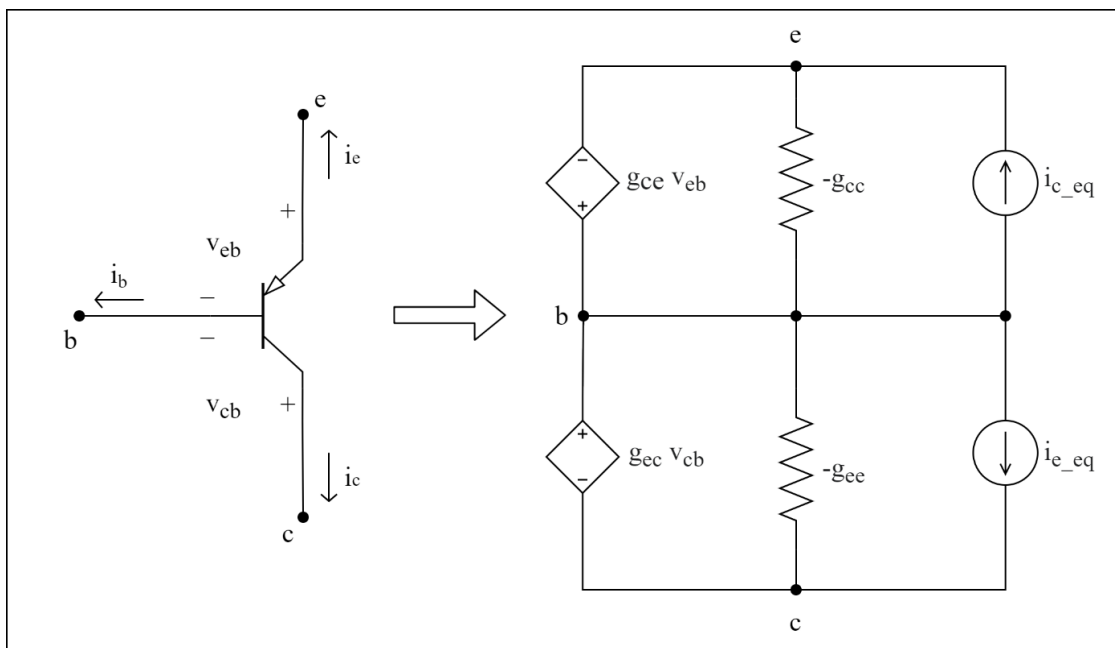
Figura 10 – Modelo de Ebers-Moll para o BJT do tipo PNP



Fonte: Produção do próprio autor.

O modelo com fontes e resistências também é similar ao definido para dispositivos NPN, com a diferença de que todas as fontes (ambas de tensão controlada e ambas de corrente) do diagrama da Figura 11 têm sua polaridade invertida.

Figura 11 – Modelo para o BJT PNP baseado em componentes lineares

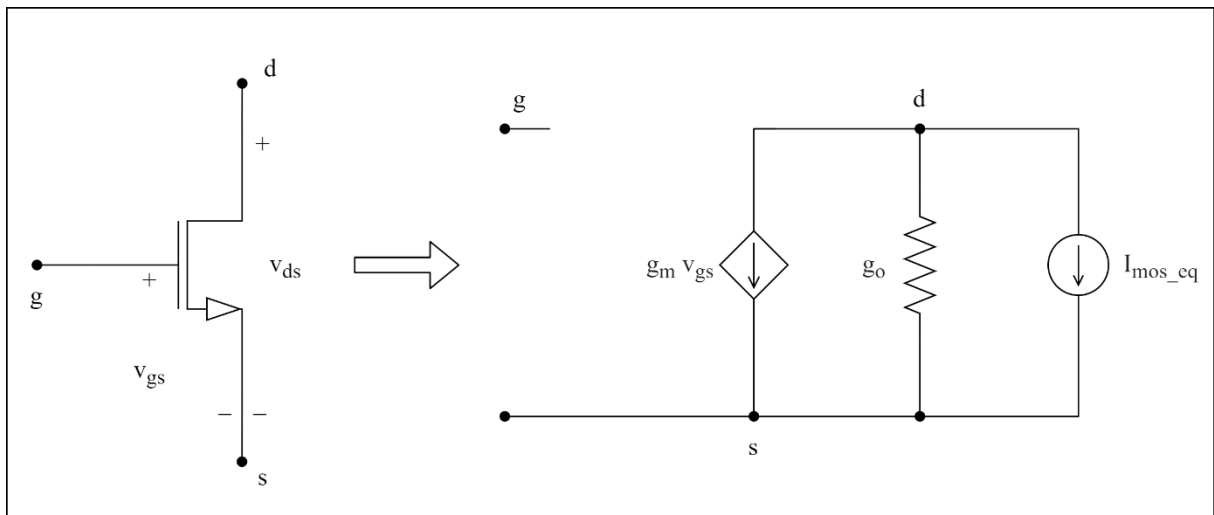


Fonte: Produção do próprio autor.

2.5.4 Modelo Linear para o Transistor de Efeito de Campo Metal-Óxido-Semicondutor

O MOSFET é um dispositivo não-linear que possui os seguintes terminais: dreno, fonte, porta e substrato (do inglês *drain*, *source*, *gate* e *bulk*, respectivamente). O substrato é comumente conectado eletricamente à fonte, sendo, portanto, omitido em algumas representações por diagrama. As relações entre as tensões entre a porta e a fonte (v_{gs}), dreno e a fonte (v_{ds}) e a tensão limiar v_{th} (considerada constante nesse trabalho) resultam em três regiões de operação distintas: corte, saturação e triodo, que serão descritas para dispositivos com canais do tipo N. A Figura 12 ilustra o modelo apresentado por Herbst e Levitt (2008) para a simulação de um MOSFET.

Figura 12 – Modelo para o MOSFET de canal tipo N baseado em componentes lineares



Fonte: Produção do próprio autor.

A região de corte ocorre quando $v_{gs} - v_{th} < 0$, que idealmente implica em nenhuma condução de corrente pelo dispositivo. Essa condição poderia ser aplicada ao modelo tornando todos os seus parâmetros iguais a zero, porém, na prática, eles são ajustados para assumir valores relativamente pequenos (da ordem de 10^{-10}), visando considerar as pequenas correntes reversas que ainda podem passar pelo MOSFET quando ele opera nessa região.

Para a região de saturação, que acontece quando $0 \leq v_{gs} - v_{th} < v_{ds}$, considera-se o Efeito Early, cuja influência pode ser escrita em função da tensão de Early (V_a).

Como primeiro passo para dedução dos termos, definem-se os termos I_{mos} (equação (38)), que representa a corrente do dreno à fonte.

$$I_{mos} = \frac{K}{2} \cdot (v_{gs} - v_{th})^2 \cdot \left(1 + \frac{v_{ds} - v_{ds,sat}}{V_a}\right) \quad (38)$$

Onde K é uma constante que representa as características construtivas do MOSFET e $v_{ds,sat} = v_{gs} - v_{th}$.

A partir desses valores, pode-se encontrar o ganho g_m e a condutância g_o do modelo, que estão definidos em (39) e (40).

$$g_m = \frac{\partial I_{mos}}{\partial v_{gs}} = K \cdot (v_{gs} - v_{th}) \cdot \left(1 + \frac{v_{ds} - v_{ds,sat}}{V_a}\right) \quad (39)$$

$$g_o = \frac{\partial I_{mos}}{\partial v_{ds}} = \frac{K}{2} \cdot \frac{(v_{gs} - v_{th})^2}{V_a} \quad (40)$$

Assim, é possível deduzir a equação (41), que define a corrente I_{mos_eq} .

$$I_{mos_eq} = I_{mos} - g_m \cdot v_{gs} - g_o \cdot v_{ds} \quad (41)$$

A terceira e última zona de operação do MOSFET é chamada de triodo, que tem seu comportamento definido pelas equações 42, 43 e 44. Nesse caso, o Efeito Early não é incluído ao modelo.

$$I_{mos} = K \cdot \left(v_{gs} - v_{th} - \frac{v_{ds}}{2}\right) \cdot v_{ds} \quad (42)$$

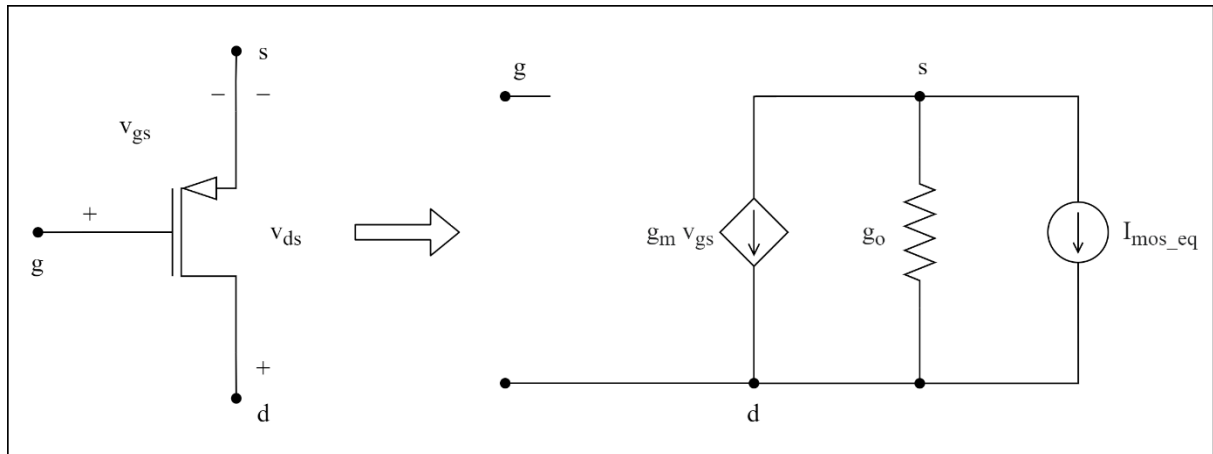
$$g_m = \frac{\partial I_{mos}}{\partial v_{gs}} = K \cdot v_{ds} \quad (43)$$

$$g_o = \frac{\partial I_{mos}}{\partial v_{ds}} = K \cdot (v_{gs} - v_{th} - v_{ds}) \quad (44)$$

A fonte de corrente nessa região também tem seu valor definido pela equação (41).

O modelo linear para dispositivos com canal P é análogo ao já obtido para MOSFETs do tipo N (Figura 13), com a distinção de que as diferenças de potencial utilizadas nas equações são v_{sg} e v_{sd} , que são os valores opostos a v_{gs} e v_{ds} , respectivamente.

Figura 13 – Modelo para o MOSFET de canal tipo P baseado em componentes lineares



Fonte: Produção do próprio autor.

2.6 Análise de Circuitos Variantes no Tempo

Os métodos abordados até aqui permitem analisar circuitos invariantes no tempo, ou seja, configurações que sempre têm o mesmo ponto de operação independente de suas condições iniciais. Circuitos com componentes que armazenam energia, como capacitores e indutores, dão origem a sistemas de equações diferenciais, cujas soluções variam de acordo com condições iniciais. Como a resolução desses sistemas de forma analítica por computadores é impraticável, simuladores como o SPICE trabalham com soluções próximas às exatas através de integrações numéricas de pontos discretizados.

Um dos métodos comumente utilizados em simuladores para obter soluções de equações diferenciais é o método de Euler implícito, um processo cujas iterações se baseiam na equação (45).

$$y_{n+1} = y_n + h \cdot f(t_{n+1}, y_{n+1}) \quad (45)$$

Em (45) e (46), y_{n+1} e y_n correspondem a aproximações da solução da equação diferencial nos tempos t_{n+1} e t_n , respectivamente. h é definido como o tamanho do passo tal que $t_{n+1} = t_n + h$, enquanto $f(t_{n+1}, y_{n+1})$ e $f(t_n, y_n)$ representam as derivadas de y em t_{n+1} e t_n , respectivamente.

Outro método também usado em simuladores é a regra trapezoidal, que é uma das fórmulas de Newton-Cotes. Seu processo iterativo está representado na equação (46).

$$y_{n+1} = y_n + h \cdot \frac{f(t_{n+1}, y_{n+1}) + f(t_n, y_n)}{2} \quad (46)$$

A escolha do método a ser aplicado ao simulador depende de vários fatores, como a precisão e estabilidade. A regra trapezoidal é mais precisa que o método de Euler implícito, porém pode ser mais instável para algumas condições. Ambos são utilizados por padrão em diferentes versões do SPICE, havendo, inclusive, a possibilidade de escolher qual deve ser aplicado. No projeto a ser executado, optou-se por utilizar a regra trapezoidal, devido à sua melhor precisão.

Como exemplo da aplicação da regra trapezoidal, encontra-se um modelo linear para o capacitor, através das duas etapas a seguir:

- Aplica-se o método de integração à equação da corrente em função da tensão para o capacitor.
- O resultado pode ser usado para encontrar um modelo baseado em componentes lineares cujas variáveis de operação podem ser calculadas pelo método da análise nodal modificada.

Primeiro, escreve-se a equação (47), que representa a relação $I \times V$ de um capacitor de capacitância C .

$$I_C = C \frac{dv_C}{dt} \quad (47)$$

Em seguida, integra-se a equação (47) em relação do tempo, resultando na equação (48).

$$v_C(t + h) = v_C(t) + \frac{1}{C} \int_t^{t+h} I_C(u) du \quad (48)$$

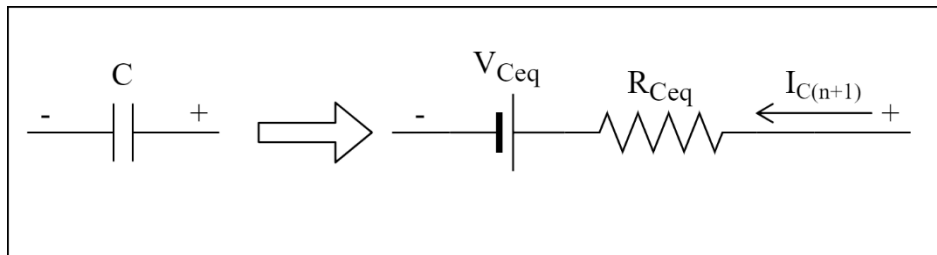
Aplica-se, então, o método trapezoidal à equação (48), resultando em (49) e, após substituição, (50). Os termos $v_{C(n+1)}$ e $v_{C(n)}$ representam as tensões nas iterações seguintes (instante de tempo $t+h$) e atual (instante de tempo t).

$$v_{C(n+1)} = v_{C(n)} + h \cdot \frac{\left[\frac{dv_C}{dt}\right]_{t+h} + \left[\frac{dv_C}{dt}\right]_t}{2} \quad (49)$$

$$v_{C(n+1)} = v_{C(n)} + \frac{h}{2C} \cdot ([I_C]_{t+h} + [I_C]_t) \quad (50)$$

Em termos de circuitos elétricos, a equação (50) pode ser interpretada como uma fonte de tensão em série com uma resistência com parâmetros mostrados na Figura 14. Chama-se $[I_C]_{t+h}$ de $I_{C(n+1)}$, que corresponde à corrente do capacitor na próxima iteração; e $[I_C]_t$ de $I_{C(n)}$, que corresponde à corrente na iteração atual.

Figura 14 – Circuito linear equivalente para o capacitor



Fonte: Produção do próprio autor.

Os valores de V_{Ceq} e R_{Ceq} são definidos pelas equações (51) e (52).

$$V_{Ceq} = v_{C(n)} + \frac{h}{2C} I_{C(n)} \quad (51)$$

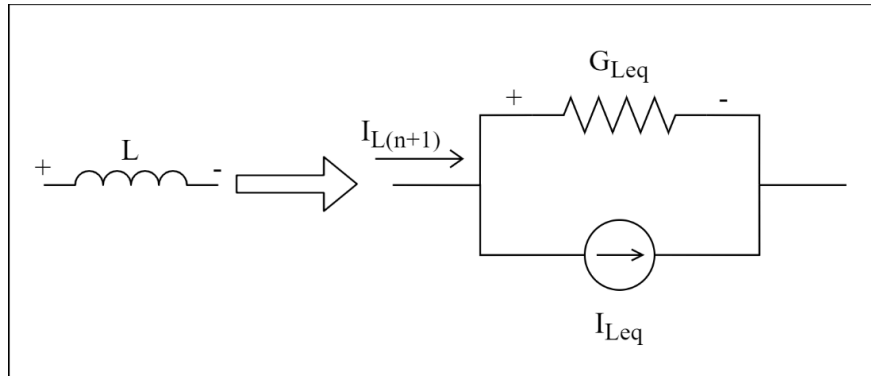
$$R_{Ceq} = \frac{h}{2C} \quad (52)$$

O valor da tensão entre os terminais do capacitor na próxima iteração ($v_{C(n+1)}$) é igual a soma de V_{Ceq} com a queda de tensão do resistor R_{Ceq} , que pode ser calculada através da Lei de Ohm com a corrente $I_{C(n+1)}$.

Realizando o processo de forma análoga para um indutor de indutância L , obtém-se o modelo mostrado na

Figura 15.

Figura 15 – Circuito linear equivalente para o indutor



Fonte: Produção do próprio autor.

Os parâmetros de corrente (I_{Leq}) e condutância (G_{Leq}) são dados pelas equações (53) e (54), respectivamente. Considera-se que $v_{L(n)}$ e $I_{L(n)}$ são os valores atuais de tensão e corrente do indutor, respectivamente.

$$I_{Leq} = I_{L(n)} + \frac{h}{2L} v_{L(n)} \quad (53)$$

$$G_{Leq} = \frac{h}{2L} \quad (54)$$

A corrente do indutor na próxima iteração ($I_{L(n+1)}$) é dada pela soma da corrente I_{Leq} com a corrente sobre a condutância G_{Leq} ($v_{L(n+1)} \cdot G_{Leq}$).

Para gerar resultados precisos, é importante que h seja suficientemente pequeno, de forma que as soluções encontradas no tempo discreto sejam próximas às que seriam obtidas caso a integração fosse feita em tempo contínuo. Em simuladores, essa variável é geralmente conhecida como *time step*, que, em tradução livre para o português, significa passo em tempo. Muitos simuladores não permitem a alteração direta desse parâmetro, cabendo ao usuário indicar apenas seu valor máximo. Nesse caso, o programa define o melhor passo em tempo para a simulação conforme as condições do circuito, podendo haver alterações neste parâmetro

durante o próprio tempo de execução (JAHN; MARGRAF; HABCHI; JACOB, 2007). Esse mecanismo é útil para circuitos não lineares, por exemplo, onde a redução do passo pode ser necessária para que se atinja a convergência durante a obtenção dos pontos de operação dos componentes. Com o objetivo de simplificar a execução do projeto, o simulador desenvolvido considera o parâmetro h constante durante todo o período de simulação do circuito, sendo o seu valor definido pelo usuário através da interface gráfica do simulador.

3 ETAPAS DE DESENVOLVIMENTO E METODOLOGIA

3.1 Etapas de Desenvolvimento

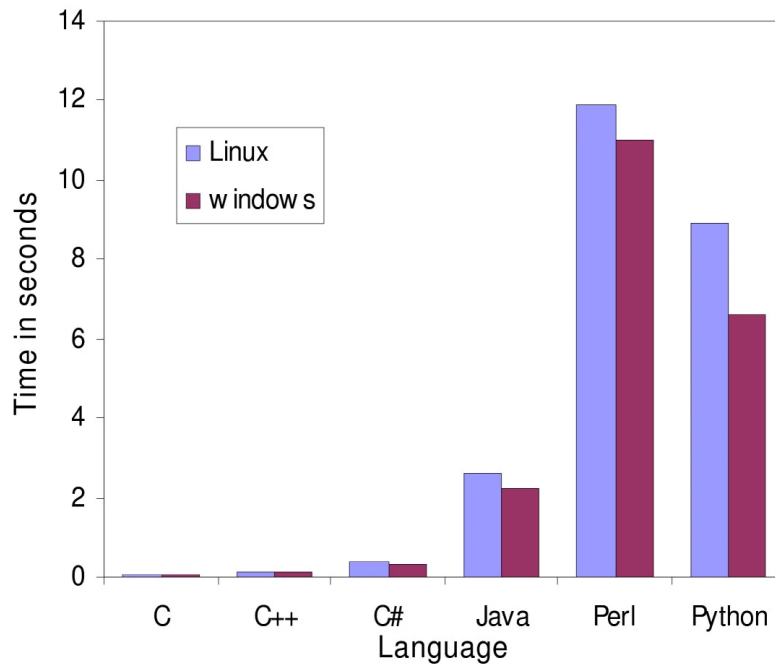
3.1.1 Linguagem de programação

Um aspecto importante deste trabalho foi a escolha da linguagem de programação para o desenvolvimento da aplicação. Durante a pesquisa, foram consideradas as linguagens C++, Java e Python, que oferecem recursos de alto nível, incluindo a implementação nativa de estruturas como listas e o gerenciamento de memória automático (FOURMENT; GILLINGS, 2008). Além disso, apresentam a vantagem de ser bastante utilizadas, implicando na disponibilidade de uma variedade de bibliotecas e *frameworks* que nelas se baseiam. Outra característica essencial dessas linguagens foi o fato de viabilizarem a utilização de programação orientada a objetos, que facilita a definição de dados e operações no programa.

Um estudo conduzido por Fourment e Gillings (2008) realizou a comparação de diversas linguagens de programação através da implementação de diferentes tipos de algoritmos computacionais na área de bioinformática. Um dos métodos incluídos na pesquisa, chamado de *Neighbor Joining*, envolve operações com matrizes, com tarefas computacionais similares às encontradas em simuladores de circuitos elétricos. O trabalho investigou o desempenho das linguagens com base em três critérios: tempo de execução; consumo de memória; e a facilidade de uso, que foi definida como o número de linhas de código necessárias para escrever um determinado programa em cada linguagem.

A Figura 16 mostra os resultados obtidos para o tempo de execução de um programa envolvendo o método *Neighbor-Joining*, nos sistemas operacionais *Windows* e *Linux*.

Figura 16 – Tempo de execução do algoritmo *Neighbor-Joining* para diferentes linguagens de programação



Fonte: Fourment e Gillings (2008).

O estudo concluiu que Python e Java apresentam a vantagem da facilidade de uso quando comparadas a C++, porém são menos vantajosas nos quesitos desempenho e consumo de memória. Considerando que uma das características mais desejáveis para um simulador de circuitos elétricos é um menor tempo de simulação, optou-se pelo uso da linguagem C++ para o desenvolvimento deste trabalho.

3.1.2 Ferramentas auxiliares

O simulador foi desenvolvido utilizando o *Microsoft Visual Studio Community Edition*. Trata-se de um ambiente de desenvolvimento integrado (IDE, do inglês *Integrated Development Environment*) gratuito que permite o desenvolvimento de aplicações em C++ em computadores com o sistema operacional *Windows*. Uma versão do simulador também foi gerada para sistemas operacionais baseados em *Linux*, sendo esta realizada com auxílio do *software Visual Studio Code*.

Para o desenvolvimento da interface gráfica do simulador, utilizou-se a biblioteca de uso livre *Simple and Fast Multimedia Library*, mais conhecida pela sigla SFML. Ela fornece uma interface simples para diversos elementos do computador, incluindo a criação de janelas, gerenciamento de eventos de mouse e teclado, e a criação de processos baseados em *OpenGL*, que é uma interface de programação de aplicações usada para computação gráfica. Outra característica importante dessa ferramenta é a possibilidade de compilar programas em vários sistemas operacionais, como *Windows* e *Linux*. Além disso, ela pode ser usada em aplicações em várias linguagens de programação, incluindo C++ (GOMILA, 2021).

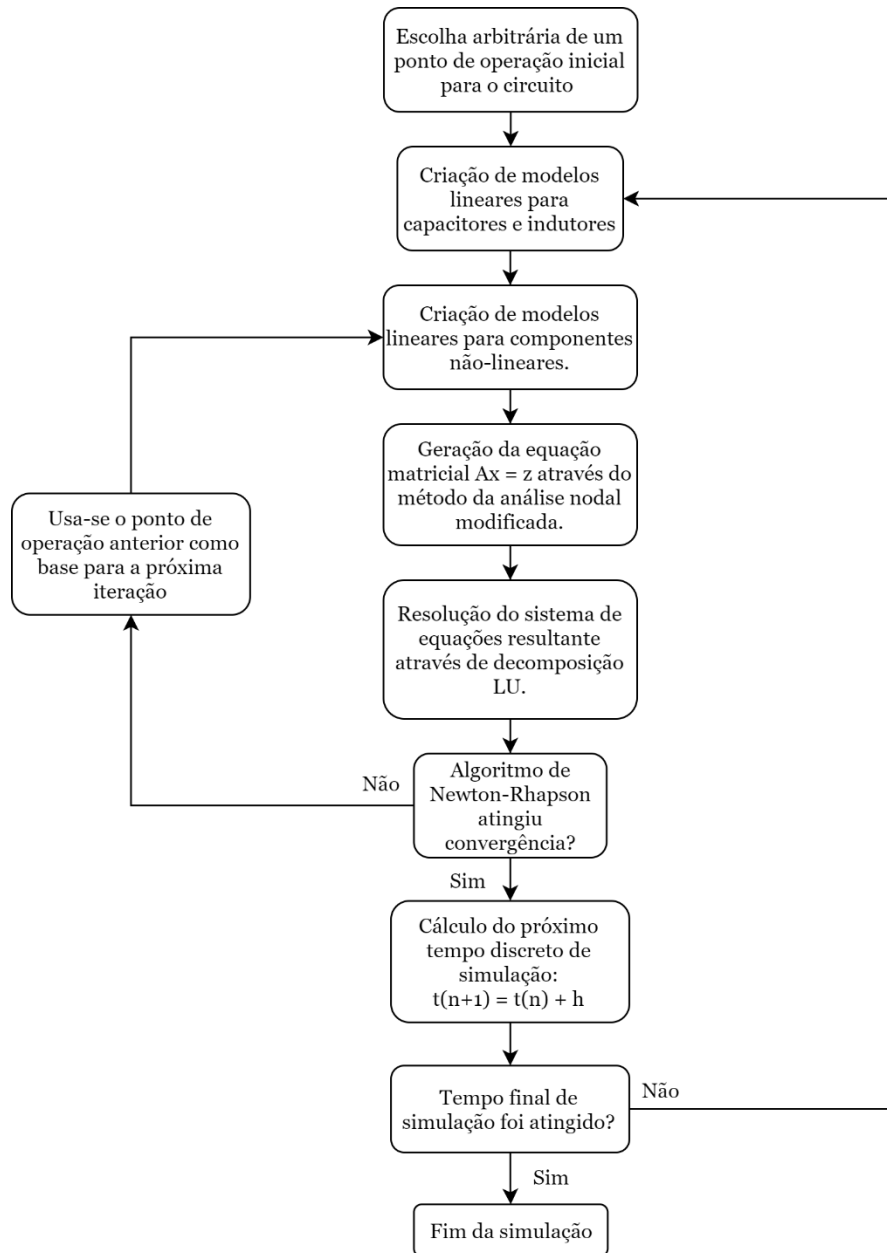
3.1.3 Biblioteca de matrizes

A primeira etapa da programação do simulador foi a criação de uma biblioteca de operações com matrizes, onde foi feita a implementação de uma classe responsável por armazenar matrizes na memória, além das rotinas que realizam a resolução de sistemas lineares. O código final inclui os métodos abordados no Referencial Teórico deste texto, como o método de Crout, substituições sucessivas e retroativas.

3.1.4 Implementação do algoritmo para análises transientes

Com base nos aspectos teóricos discutidos na seção de Referencial Teórico e no diagrama produzido por Lannutti, Nenzi e Olivieri (2012), pode-se criar o diagrama da Figura 17, que descreve o algoritmo básico do simulador.

Figura 17 – Diagrama do algoritmo do simulador



Fonte: Lannutti, Nenzi e Olivieri (2012).

Nota: Adaptado pelo autor.

A primeira tarefa desta etapa do trabalho consistiu na criação de uma classe que descrevesse as conexões dos terminais de um componente, que é uma característica comum a todos os elementos que podem ser simulados no programa. A implementação foi feita através de uma lista que associa cada terminal ao nó ao qual ele está conectado, cujo número representante é o mesmo que o utilizado para a análise nodal de circuitos.

A etapa seguinte consistiu em realizar a representação em código de resistores e fontes de tensão e corrente independentes. A partir das informações desses componentes básicos, que estão presentes na maioria dos modelos para os elementos simuláveis, pôde-se implementar o algoritmo de análise nodal modificada.

Na terceira etapa, adicionou-se os modelos para indutores e capacitores à aplicação, que foram obtidos com base na regra trapezoidal para a aproximação das soluções de equações diferenciais. Nessa fase, também foi acrescentado ao programa o *loop* principal da simulação transiente, a partir da entrada de valores de *step* e tempo total.

A quarta etapa consistiu na introdução dos componentes não lineares ao programa. Para isso, foi necessário implementar o método de Newton-Raphson, tarefa realizada em paralelo com adição de diodos. Em seguida, MOSFETs e BJTs foram incluídos ao simulador. Visando garantir a convergência para alguns tipos de topologias, foram necessárias algumas alterações no algoritmo e nos parâmetros do método de Newton-Raphson, para manter uma estabilidade satisfatória considerando erros de arredondamento e truncamento. Primeiro, ajustou-se os valores de *reltol* e *vntol*, de forma a diminuir o número de iterações necessárias para o término do ciclo. Embora essa modificação tenha sido suficiente para a maioria dos casos de teste, alguns circuitos também exigiram a aplicação do algoritmo mostrado na Figura 6.

Para que o usuário pudesse analisar os formatos de tensões/correntes no circuito, decidiu-se adicionar à lógica da aplicação o conceito de voltímetros e amperímetros. Essa solução é a mesma implementada pelo simulador PSIM, e tem a vantagem de ser intuitiva para o usuário, visto que é análoga à forma como medições são realizadas em circuitos reais.

A implementação de voltímetros foi simples, uma vez que a solução da equação (5) já fornece as tensões em todos os nós do circuito, tendo sido necessário apenas encontrar a diferença de potencial entre os nós conectados aos terminais dos medidores.

A inclusão de amperímetros não pôde ser realizada de forma direta, pois o método de análise nodal modificada não calcula as correntes para cada ramo em um circuito. A solução encontrada para modelar esse componente baseou-se no procedimento realizado para fontes controladas por corrente, que acrescentam uma fonte de tensão contínua independente de valor nulo ao ramo

cuja corrente deve ser medida. Esse processo não altera o comportamento do circuito, mas permite que a grandeza desejada seja obtida diretamente através da equação (5), já que todas as fontes de tensão independentes têm suas correntes calculadas pelo método de análise nodal modificada. Portanto, cada amperímetro colocado no circuito é associado a uma fonte de tensão de valor zero, que é conectada entre os terminais que serão medidos.

3.1.5 Interface gráfica para montagem de circuitos

O *software* produzido até então não apresentava uma interface gráfica, somente sendo possível simular circuitos descritos em arquivos de texto por linha de comando, de forma similar às primeiras versões do SPICE. Essa forma de interação com o usuário, embora simples de ser implementada, torna a construção de circuitos uma tarefa dispendiosa e propensa a erros, especialmente para diagramas grandes e com muitos nós.

Optou-se pelo desenvolvimento de uma interface simples, com uma área para desenho dos diagramas de circuitos, e uma barra lateral para a configuração dos parâmetros dos componentes e da simulação em geral. Esse *layout* foi influenciado pelo programa *Falstad Circuit Simulator*, mas apresenta algumas diferenças, como a forma de modificação dos valores dos componentes, que é feita em uma barra lateral, ao invés de janelas de *pop up* que aparecem para o usuário quando o elemento é selecionado.

Os controles do simulador, isto é, a forma como itens são adicionados, movidos e rotacionados na área de desenho, foi influenciada pelo simulador PSIM, que apresenta controles simples e intuitivos. Alguns atalhos de adição de componentes, por exemplo, foram usados no simulador, para facilitar a adaptação de usuários já habituados àquela ferramenta.

Também foi adicionado à aplicação uma barra de menus na parte superior da janela da tela, um item muito comum em vários programas de computador. Nessa etapa de desenvolvimento, foram adicionadas diversas opções essenciais ao simulador, como a possibilidade de salvar circuitos, assim como a de abrir arquivos já salvos. Nessa barra também foi adicionado o menu que permite ao usuário escolher o elemento que deseja adicionar ao circuito.

3.1.6 Interface gráfica para exibição dos resultados

Ao final da simulação de um circuito, os resultados são armazenados em listas de valores, sendo que uma contém os pontos que definem o tempo, e outras estão relacionadas aos objetos de medição, que podem ser tensões e/ou correntes.

A primeira forma de exibição limitou-se à exibição dos pontos através da interface de linhas de comando, porém esse método dificultava a análise dos resultados, especialmente para circuitos grandes e com muitos pontos discretos simulados. Portanto, para que os produtos das simulações pudessem ser visualizados pelo usuário de uma forma mais convencional, optou-se pela criação de uma interface gráfica simples que pudesse exibir gráficos a partir de uma lista associada ao eixo das abscissas, e outras referentes às ordenadas.

Para complementar a interface, foram implementadas algumas funções básicas que facilitam a análise dos gráficos, como mover (do inglês, *pan*), ampliação e redução de tamanho (do inglês, *zoom in* e *zoom out*) e retorno à escala inicial. A programação foi amplamente baseada em técnicas de álgebra linear, com a utilização de técnicas de translação e escala para realizar a transformação dos pontos.

Outro aspecto importante da ferramenta é a possibilidade de incluir vários conjuntos de pontos em um mesmo gráfico, o que facilita a comparação das variáveis de mesma grandeza. No contexto do simulador, as medições dos voltímetros são desenhadas em um gráfico, enquanto as correntes medidas pelos amperímetros aparecem em outro. Para facilitar a diferenciação dos formatos de tensão e corrente, foi implementada uma legenda similar à utilizada em *plots* do programa MATLAB. As descrições colocadas nesse item são os nomes atribuídos aos voltímetros e amperímetros na área de montagem de circuitos.

Outras ferramentas essenciais adicionadas ao subprograma foram a indicação das coordenadas associadas ao ponteiro do mouse e a mudança automática de escala dos rótulos dos gráficos.

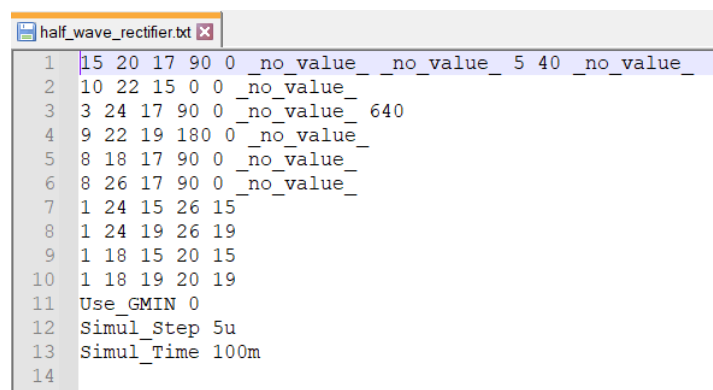
3.1.7 Exportação dos resultados para arquivos CSV

O uso da interface gráfica do próprio programa como forma de exibir os resultados é interessante para simulações rápidas, onde deseja-se obter apenas alguns pontos dos gráficos de saída. No entanto, caso seja necessário analisar o comportamento do circuito de forma mais detalhada, ou até exibi-lo de outra forma, a ferramenta pode não ser suficiente. Para satisfazer essa necessidade, foi implementada a capacidade de salvar os resultados dos circuitos em um arquivo CSV (do inglês, *Comma-Separated Values*). Esse formato possibilita que o usuário examine os resultados em aplicações mais especializadas em manipulação de dados, como *Microsoft Excel* e *LibreOffice*.

3.1.8 Armazenamento de circuitos em arquivos de texto

Um dos atributos mais importantes de qualquer *software* de simulação é a possibilidade de salvar as topologias criadas durante a execução, para que possam ser modificadas e/ou utilizadas posteriormente. Foi, portanto, desenvolvido um protocolo para transformar os componentes desenhados na área de montagem de circuitos e os parâmetros de simulação em vetores de informação (*strings*) que podem ser armazenados em arquivos de texto. A Figura 18 mostra um exemplo de aplicação desse protocolo, para um circuito retificador de meia onda.

Figura 18 – Arquivo de texto para retificador de meia onda



```

1 |15 20 17 90 0 _no_value_ _no_value_ 5 40 _no_value_
2 |10 22 15 0 0 _no_value_
3 |3 24 17 90 0 _no_value_ 640
4 |9 22 19 180 0 _no_value_
5 |8 18 17 90 0 _no_value_
6 |8 26 17 90 0 _no_value_
7 |1 24 15 26 15
8 |1 24 19 26 19
9 |1 18 15 20 15
10 |1 18 19 20 19
11 |Use_GMIN 0
12 |Simul_Step 5u
13 |Simul_Time 100m
14

```

Fonte: Produção do próprio autor.

3.2 Metodologia

3.2.1 Classificação da pesquisa

Com base nas classificações propostas por Prodanov e Freitas (2013) com relação aos tipos de pesquisa científica, pode-se afirmar que, do ponto de vista de sua natureza, o projeto é uma pesquisa aplicada, pois tem como objetivo gerar conhecimentos voltados para a solução de problemas específicos, neste caso, o comportamento transitório de circuitos elétricos. Com relação a seus objetivos, o projeto pode ser classificado como uma pesquisa exploratória, pois visa coletar mais informações sobre seu objeto de estudo. Considerando os procedimentos técnicos, é um estudo de caso, pois envolve um estudo aprofundado e minucioso do funcionamento de simuladores já existentes. A abordagem utilizada será baseada no método dedutivo, pois parte dos princípios e leis consideradas verdadeiras em circuitos elétricos para modelar os componentes eletrônicos que serão incluídos ao programa final.

3.2.2 Avaliação do desempenho do simulador

Como forma de avaliar o simulador a ser desenvolvido, analisou-se alguns fatores esperados de um bom simulador, com base nos aspectos propostos por Ngada (2014), já citados na Seção 1.2.1.

A avaliação da robustez de um simulador foi realizada por meio da comparação de seus resultados com aqueles de programas de simulação já conhecidos na área da engenharia elétrica. Os fatores examinados foram a precisão dos resultados e a usabilidade de cada *software*. Os programas de simulação selecionados para as comparações foram o LTspice XVII, que é baseado em SPICE, e o QUCS, que utiliza um núcleo de simulação próprio. Assim, o desempenho do simulador pôde ser avaliado com base em programas que usam métodos distintos para alcançar o mesmo objetivo.

O primeiro critério avaliado foi a precisão do simulador desenvolvido. Para isso, foram comparados os resultados para três topologias de circuitos elétricos: um circuito RLC série, um retificador de onda completa com filtro capacitivo e um seguidor de emissor. Em todos os simuladores, os parâmetros dos componentes foram regulados com os mesmos valores, ou o

mais próximo possível, no caso de elementos não lineares. As condições de simulação também foram ajustadas para garantir a maior similaridade realizável entre as aplicações. O LTspice e QUCS não permitem alterar diretamente o *time step* para a simulação e, portanto, apenas seu valor máximo foi especificado, com o mesmo valor utilizado como passo para o simulador desenvolvido.

Após a realização das simulações, os resultados das três aplicações foram exportados para arquivos no formato CSV, a partir dos quais foi realizada uma comparação através de um *script* escrito na linguagem *Python*. O processo consistiu em calcular os erros (diferenças) entre as formas de ondas geradas pelo simulador desenvolvido e as obtidas pelos dois programas de referência.

Como os dois simuladores de referência têm um *time step* adaptativo, o número de pontos resultantes das simulações é sempre maior ou igual à quantidade produzida por aquela com passo constante. Os valores gerados pelo QUCS são interpolados internamente, de modo que pontos de saída já são armazenados com o intervalo de tempo constante especificado, porém o LTspice não realiza esse processo. Logo, para viabilizar uma comparação ponto a ponto das simulações, os resultados do LTspice foram interpolados linearmente, de forma que seus valores pudessem ser alinhados aos produtos das outras simulações.

As variáveis utilizadas para quantificar o erro (equação (55)) foram o erro absoluto máximo (equação (56)) e o erro médio absoluto (equação (57)). A última é baseada no livro de Hyndman e Athanasopoulos (2018), que aborda medição de precisão de modelos.

$$Erros = v_{simul\ próprio} - v_{simul\ ref} \quad (55)$$

$$EA_{máximo} = \max(|Erros|) \quad (56)$$

$$EA_{médio} = \frac{\sum_{i=1}^n |Erros_i|}{n} \quad (57)$$

Onde $v_{simul\ próprio}$ e $v_{simul\ ref}$ são o conjunto de pontos obtidos pelas aplicações desenvolvidas e de referência para uma determinada variável de medição, respectivamente. Na equação (57), n se refere ao número total de pontos analisados.

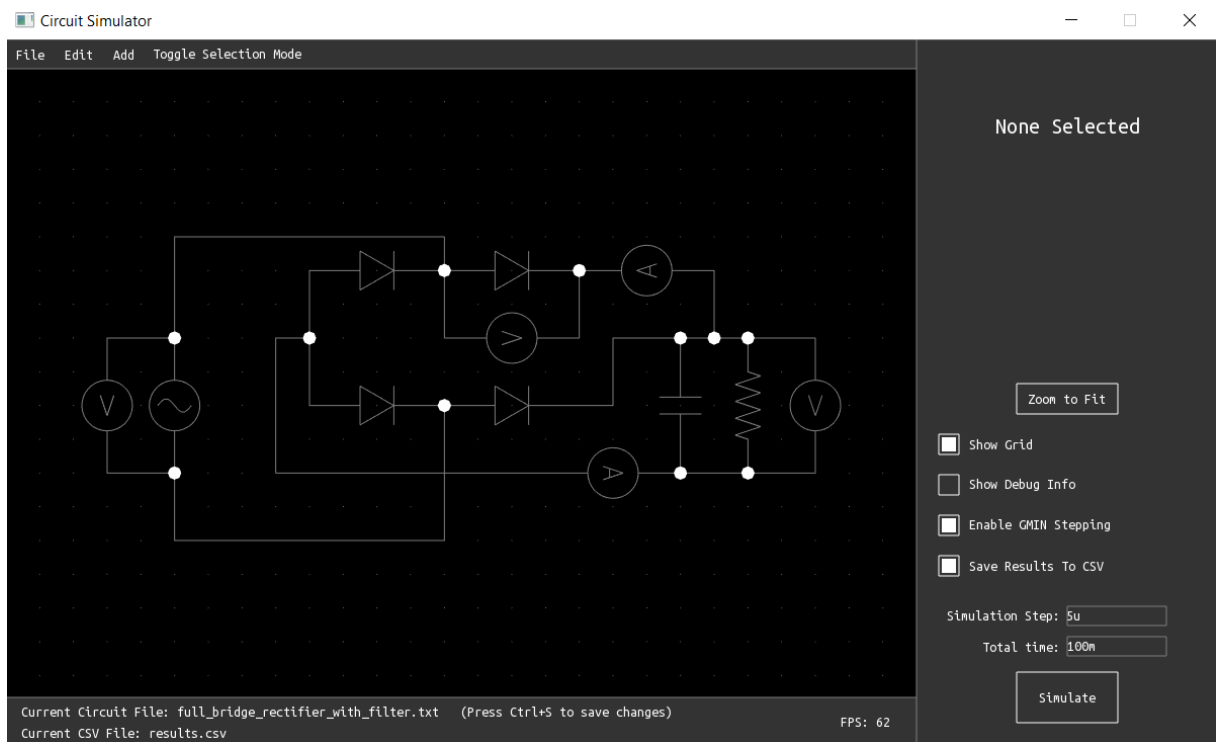
Para facilitar a interpretação dos resultados, as variáveis de erro foram comparadas com os valores absolutos máximos registrados para a respectiva grandeza medida (V_{absmax}), permitindo encontrar uma proporção relativa de erro para cada conjunto de pontos.

4 RESULTADOS E DISCUSSÃO

4.1 Interface Gráfica

A Figura 19 mostra a tela inicial da aplicação após sua inicialização. O circuito inicial é um retificador de onda completa com filtro capacitivo, que faz parte de alguns exemplos incluídos ao programa, os quais podem ser acessados por meio do menu *File*, seguido da opção *Load Circuit*.

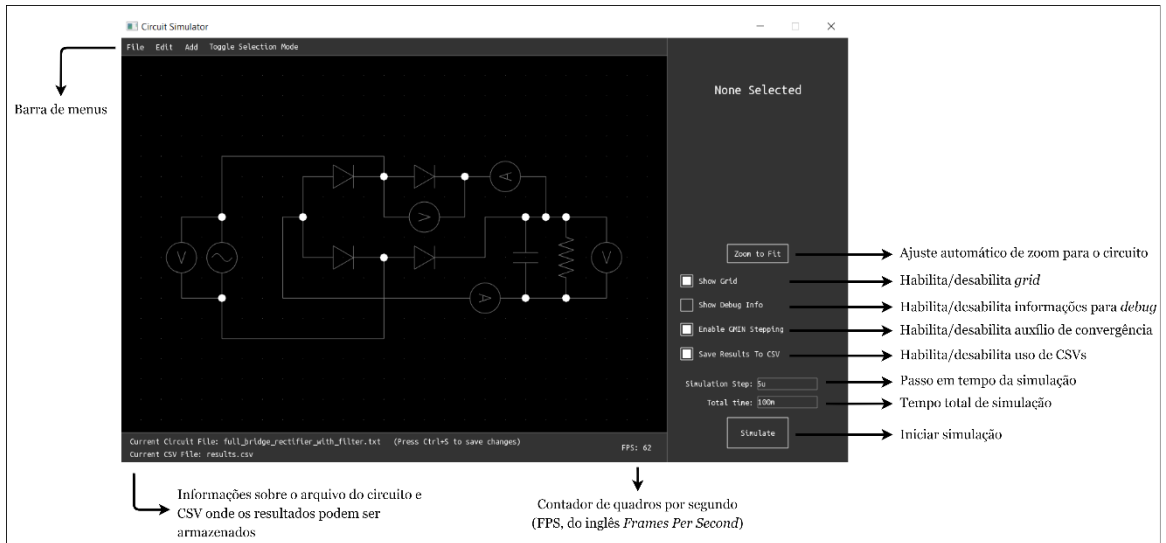
Figura 19 – Janela inicial do simulador



Fonte: Produção do próprio autor.

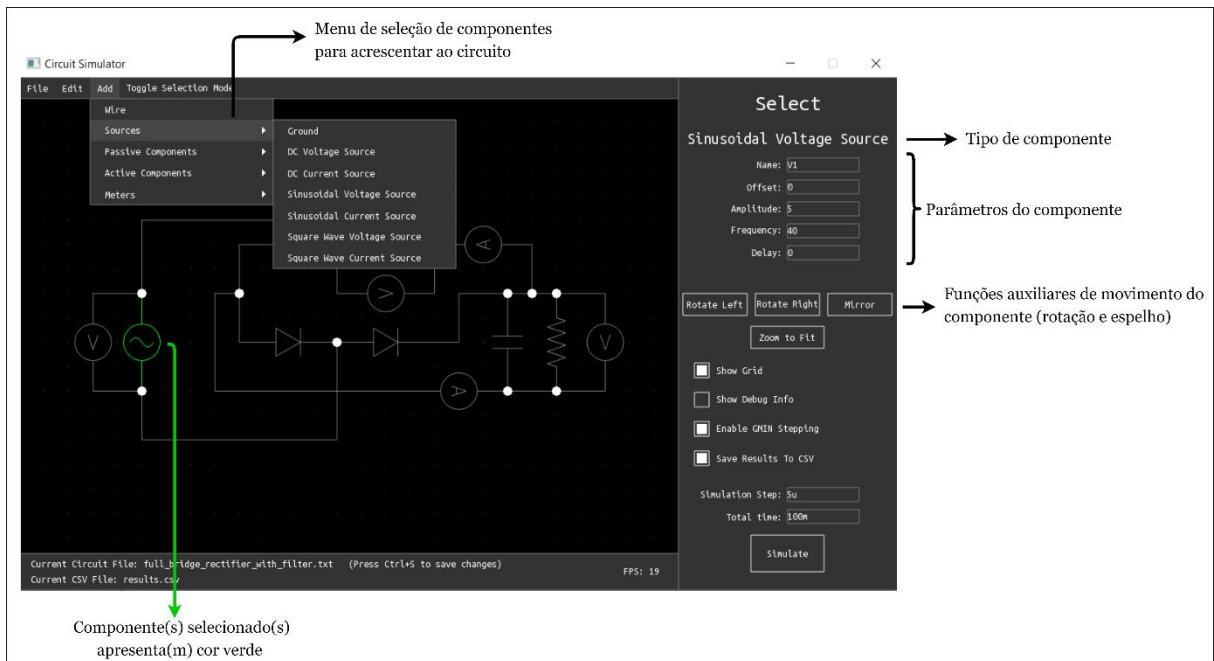
As Figuras 20 e 21 apresentam as descrições dos elementos básicos da janela principal. A adição de um componente pode ser realizada através do menu *Add*, através de um clique no item correspondente. Após colocar o elemento na posição desejada, o usuário pode modificar seus parâmetros através da barra lateral. O modo de seleção, acessível através do botão *Toggle Selection Mode* na barra de menus, também permite alteração da posição e parâmetros dos componentes já acrescentados.

Figura 20 – Descrição dos elementos básicos da interface do simulador



Fonte: Produção do próprio autor.

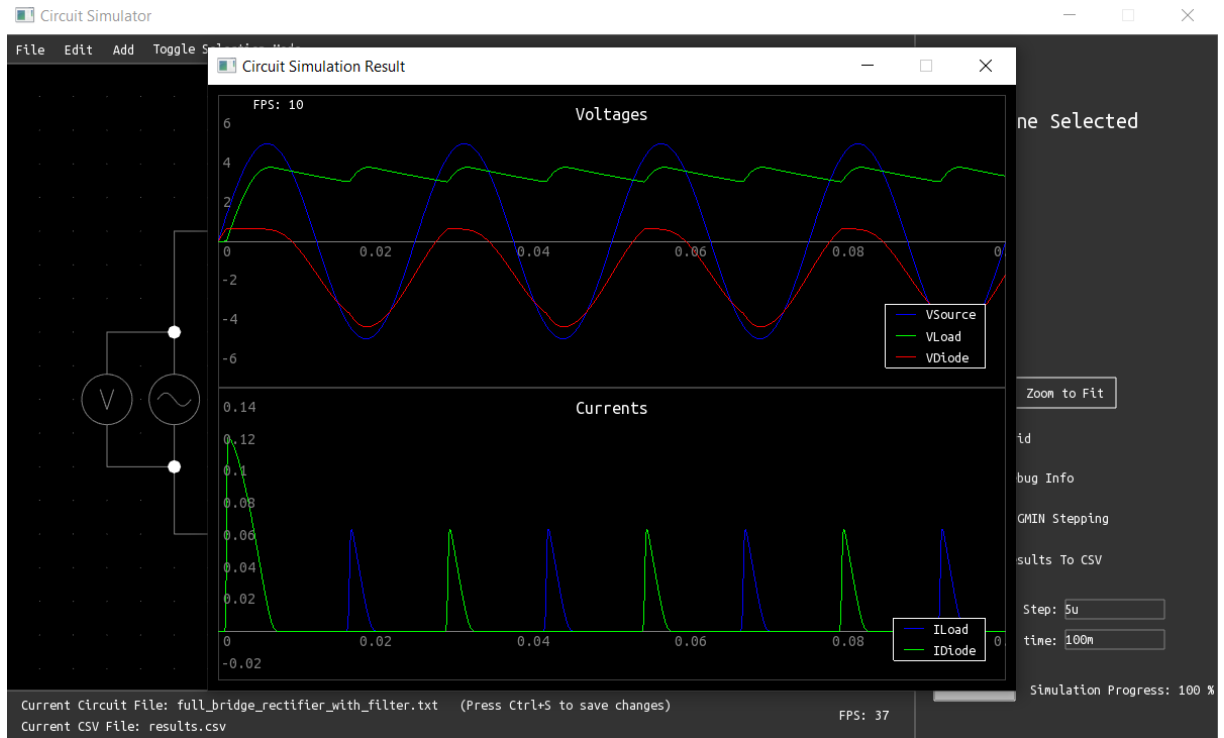
Figura 21 – Descrição de elementos usados para a montagem de diagramas



Fonte: Produção do próprio autor.

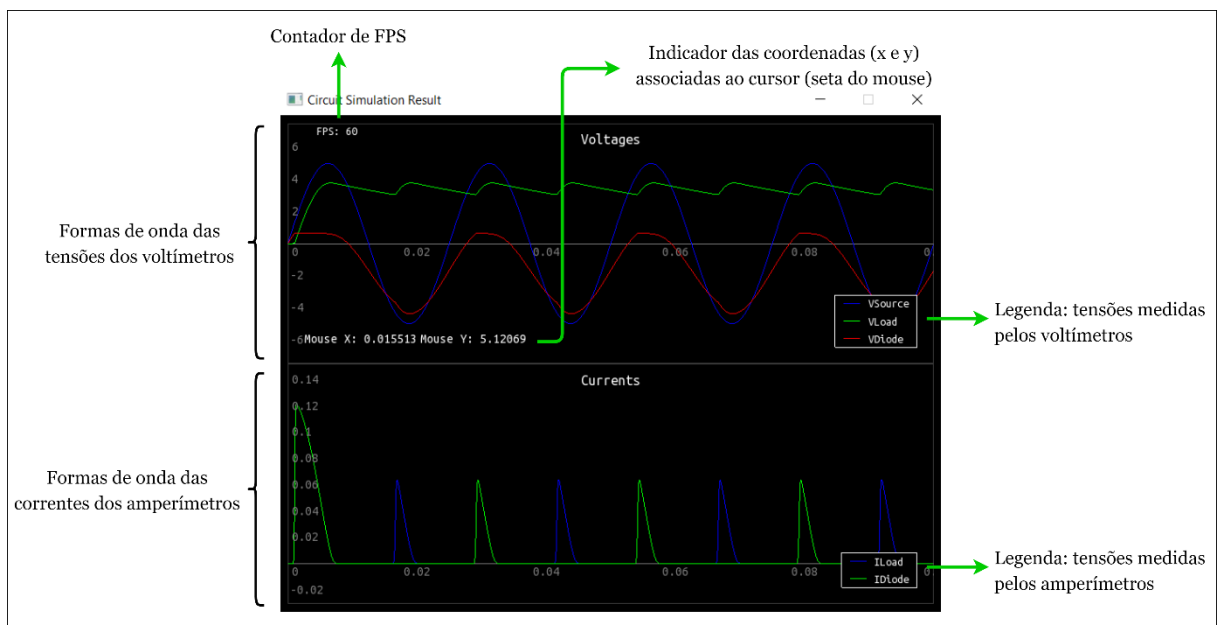
Ao clicar no botão *Simulate*, a simulação do circuito é realizada e, ao final, uma janela de exibição dos gráficos dos resultados é aberta para o usuário, como mostra a Figura 22. Os elementos principais da janela estão descritos na Figura 23.

Figura 22 – Janela de resultados da simulação



Fonte: Produção do próprio autor.

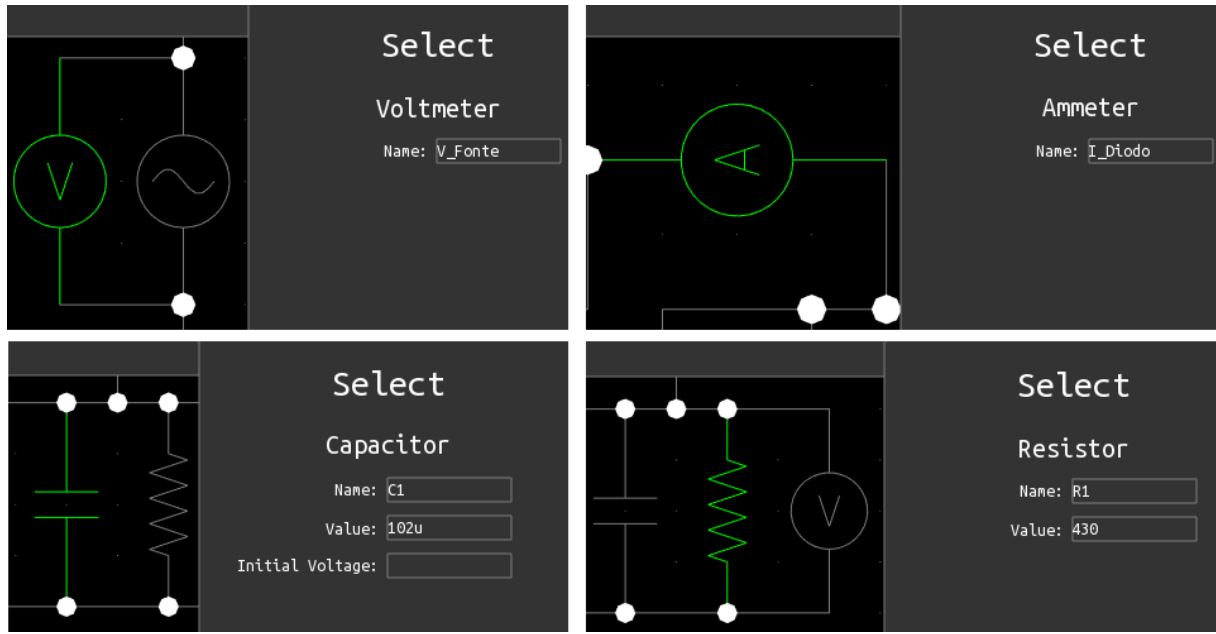
Figura 23 – Elementos da janela de resultados de simulação



Fonte: Produção do próprio autor.

Os nomes de cada curva na legenda do gráfico são determinados pelo nome do respectivo elemento de medição. A Figura 24 exemplifica a definição dos parâmetros na barra lateral para alguns componentes.

Figura 24 – Exemplos configuração de parâmetros



Fonte: Produção do próprio autor.

4.2 Desempenho do Simulador

4.2.1 Precisão dos resultados

Para avaliar a precisão das simulações, foram avaliados os resultados para três topologias elétricas diferentes, que foram comparados com as respostas dos simuladores LTspice e QUCS. Em todos os circuitos simulados, os componentes lineares (resistores, capacitores, indutores, fontes de tensão) são ideais, ou seja, não apresentam nenhuma perda quantificada diretamente pelos programas. Para os componentes não lineares (diodos, transistores bipolares de junção e MOSFETs), os parâmetros foram ajustados de forma que os circuitos nos três programas fossem o mais similar possível.

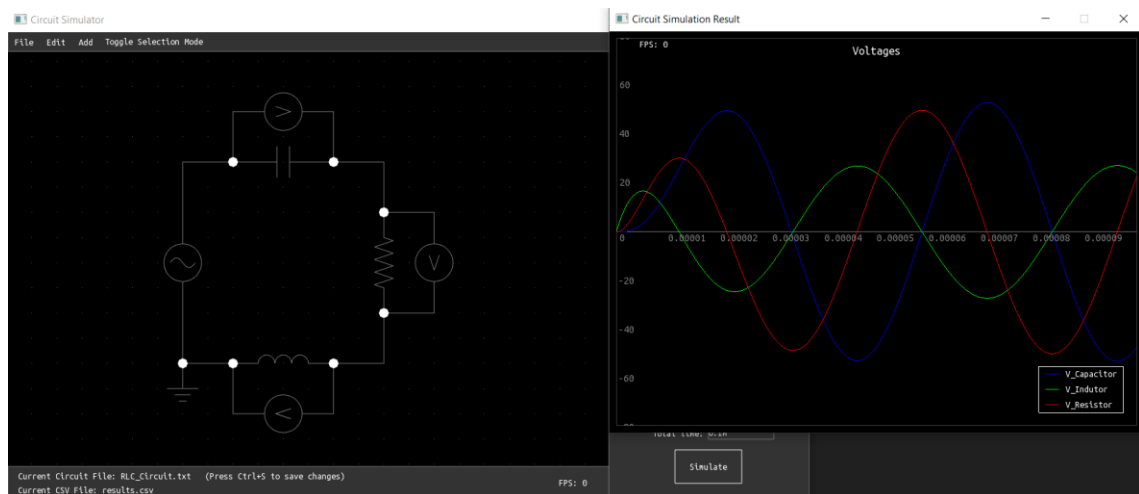
O primeiro circuito analisado foi um circuito RLC série com uma fonte de tensão senoidal. Essa configuração foi utilizada para realizar os primeiros testes durante o desenvolvimento do simulador, visto que é composta apenas de componentes lineares e, portanto, a solução da

equação matricial (5) para um determinado instante de tempo é imediata e não exige aproximação por meio do algoritmo de Newton-Raphson.

O circuito consistiu de uma fonte de tensão senoidal com amplitude de 56 V e frequência de 20 kHz, um capacitor de $0,25 \mu\text{F}$, um resistor de 30Ω , e um indutor de $130 \mu\text{H}$. O passo em tempo (máximo valor, no caso do LTspice e QUCS) escolhido foi de 5 ns, enquanto a duração total da simulação foi configurada para $100 \mu\text{s}$.

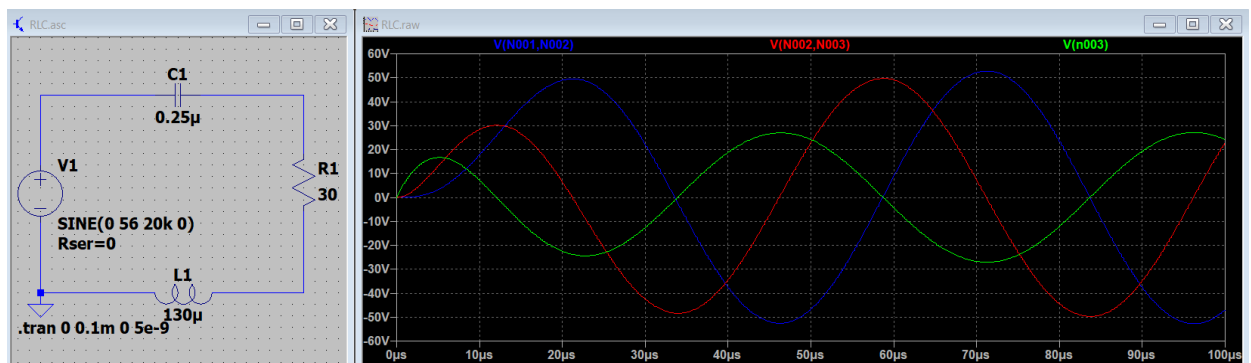
As Figuras 25, 26 e 27 mostram o esquema do circuito montado nos simuladores próprio, LTspice e QUCS, respectivamente. Os resultados, na forma como aparecem na interface gráfica dos programas, foram incluídos nas figuras.

Figura 25 – Diagrama e resultados para o primeiro circuito no simulador desenvolvido



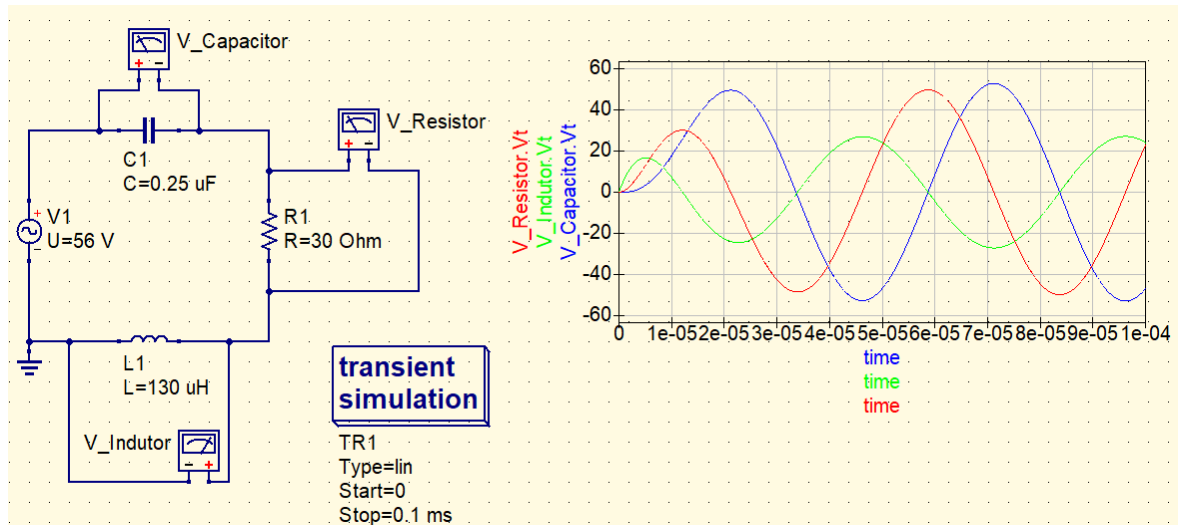
Fonte: Produção do próprio autor.

Figura 26 – Diagrama e resultados para o primeiro circuito no LTspice



Fonte: Produção do próprio autor.

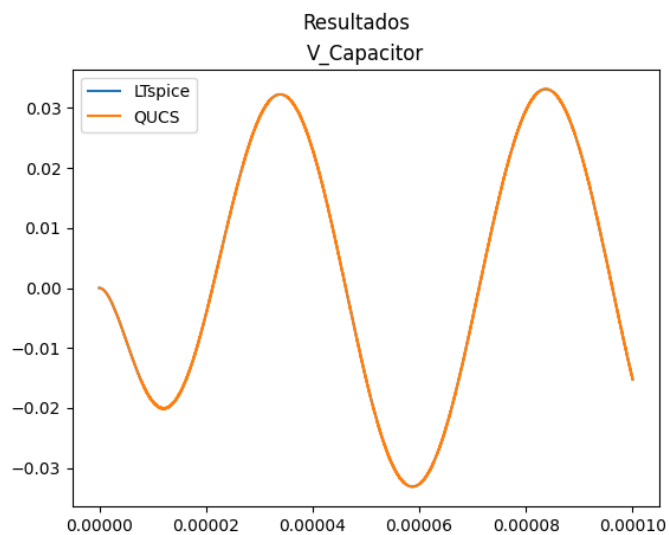
Figura 27 – Diagrama e resultados para o primeiro circuito no QUCS



Fonte: Produção do próprio autor.

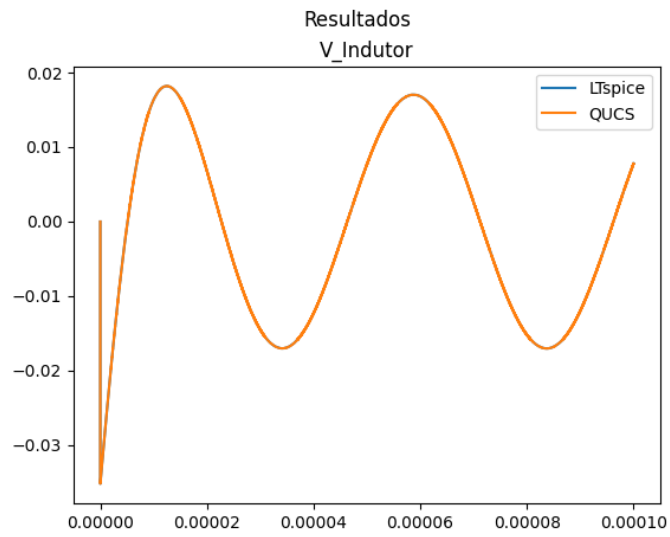
A partir dos resultados das três aplicações, foram gerados os Gráficos 2, 3 e 4, que quantificam os desvios do simulador desenvolvido em relação àqueles de referência. É importante observar que os formatos relacionados ao LTspice e QUCS são muito similares para a maioria dos objetos de medição, indicando que os resultados de ambos os programas são relativamente próximos.

Gráfico 2 – Erros obtidos para a tensão sobre o capacitor no primeiro circuito



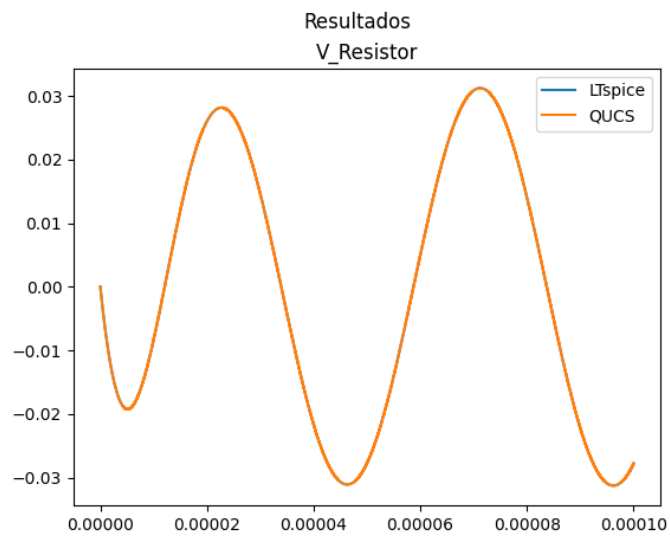
Fonte: Produção do próprio autor.

Gráfico 3 – Erros obtidos para a tensão sobre o indutor no primeiro circuito



Fonte: Produção do próprio autor.

Gráfico 4 – Erros obtidos para a tensão sobre o resistor no primeiro circuito



Fonte: Produção do próprio autor.

Para quantificar os erros observados nos gráficos, foram calculadas as variáveis das equações já abordadas na Seção 3.2.2. Os resultados encontram-se nas Tabelas 12 e 13.

Tabela 12 – Variáveis de erro calculadas em relação ao LTspice para o primeiro circuito

Variável	V_Capacitor (V)	V_Indutor (V)	V_Resistor (V)
EA _{máximo}	3.32e-02	3.52e-02	3.13e-02
EA _{médio}	1.84e-02	1.10e-02	1.89e-02
V _{absmax}	5.28e+01	2.71e+01	4.98e+01
% EA _{máximo}	0.0628	0.13	0.0628
% EA _{médio}	0.0349	0.0407	0.0381

Fonte: Produção do próprio autor.

Tabela 13 – Variáveis de erro calculadas em relação ao QUCS para o primeiro circuito

Variável	V_Capacitor (V)	V_Indutor (V)	V_Resistor (V)
EA _{máximo}	3.32e-02	3.52e-02	3.13e-02
EA _{médio}	1.84e-02	1.10e-02	1.89e-02
V _{absmax}	5.28e+01	2.71e+01	4.98e+01
% EA _{máximo}	0.0628	0.13	0.0628
% EA _{médio}	0.0349	0.0407	0.0381

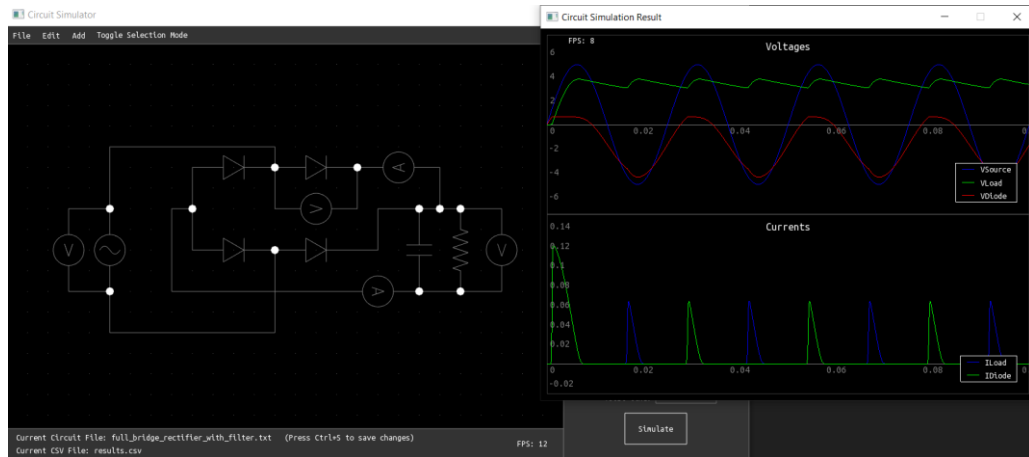
Fonte: Produção do próprio autor.

O segundo circuito avaliado foi um retificador de onda completa com filtro capacitivo. Essa topologia foi escolhida porque possibilita averiguar o comportamento de um circuito com mais de um componente não linear (diodo), além de possuir um elemento armazenador de energia (capacitor). Seu estudo também é interessante porque os diodos operam nos dois modos possíveis – condução e corte – que são modelados com base em apenas uma curva I_xV .

O circuito é composto por uma fonte de tensão senoidal de 40 Hz com tensão de pico de 5 V, um capacitor de 102 μ F, uma resistência de 430 Ω , e 4 diodos iguais com corrente de saturação igual a 1e-12 A, temperatura de 27 °C e demais parâmetros iguais ao padrão do SPICE. As configurações de simulação foram: *time step* de 5 μ s; e tempo total de simulação de 100 ms.

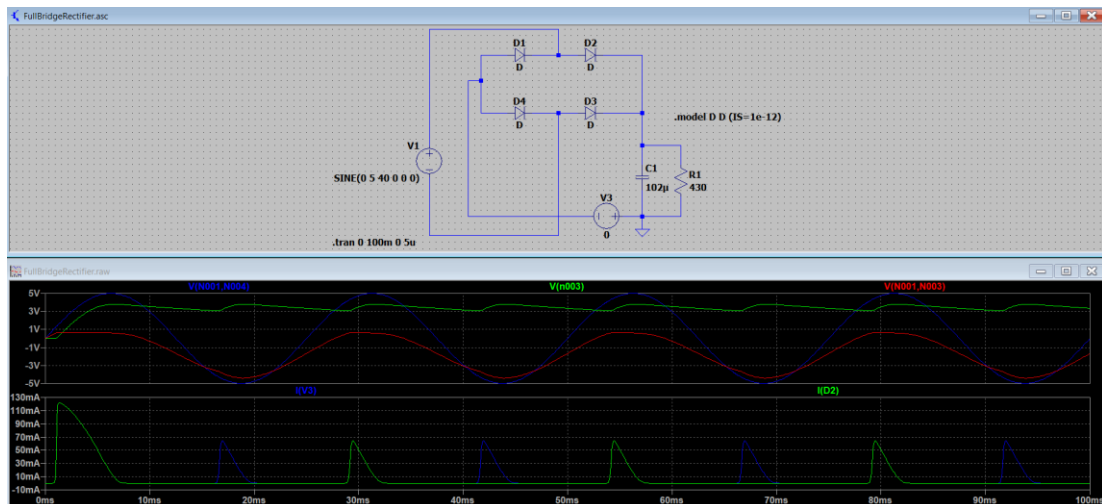
As Figuras 28, 29 e 30 mostram os esquemas de montagem nos três simuladores, assim como os respectivos resultados de cada simulação.

Figura 28 – Diagrama e resultados para o segundo circuito no simulador desenvolvido



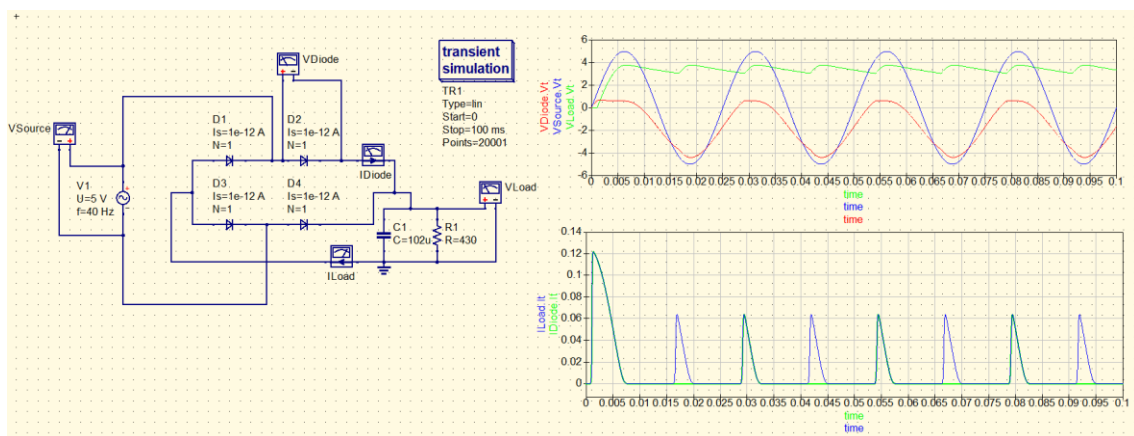
Fonte: Produção do próprio autor.

Figura 29 – Diagrama e resultados para o segundo circuito no LTspice



Fonte: Produção do próprio autor.

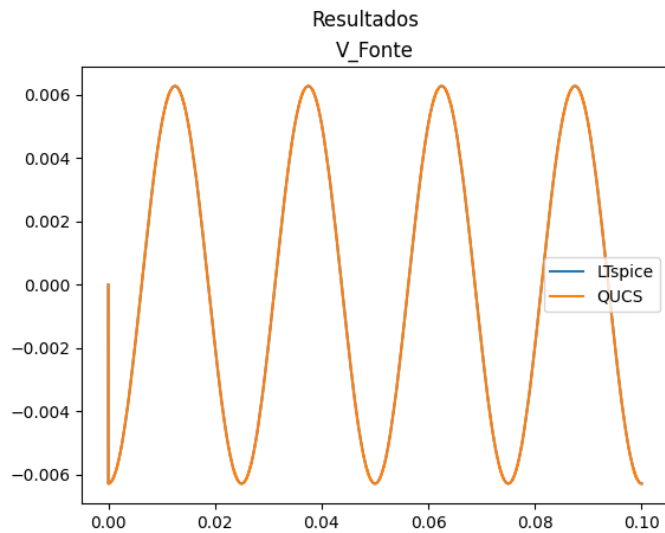
Figura 30 – Diagrama e resultados para o segundo circuito retificador no QUCS



Fonte: Produção do próprio autor.

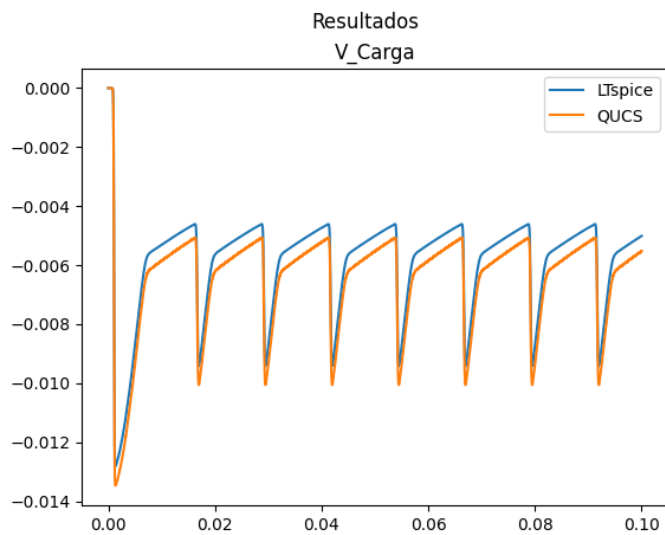
A partir dos resultados obtidos, foi possível gerar gráficos das diferenças em relação aos programas de referência para todas as grandezas medidas nas simulações, como mostram os Gráficos 5, 6, 7, 8 e 9.

Gráfico 5 – Erros obtidos para a tensão sobre a fonte de tensão no segundo circuito



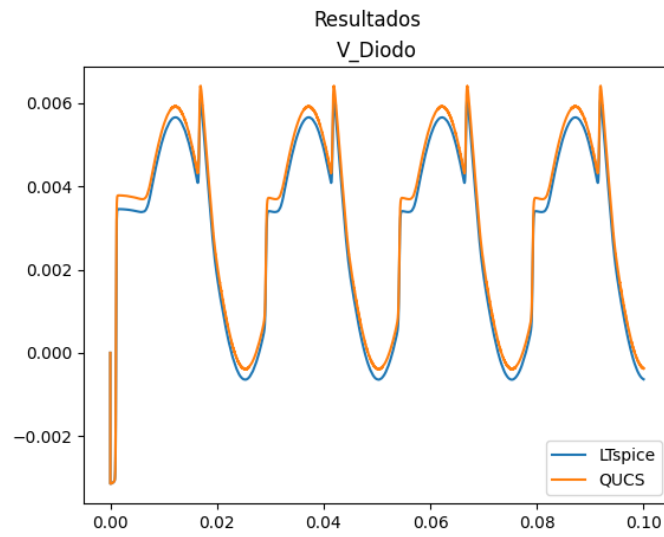
Fonte: Produção do próprio autor.

Gráfico 6 – Erros obtidos para a tensão sobre a carga no segundo circuito



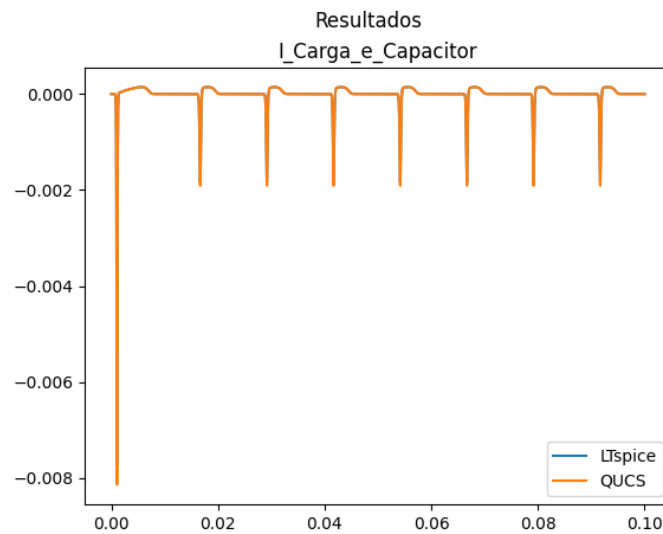
Fonte: Produção do próprio autor.

Gráfico 7 – Erros obtidos para a tensão sobre o diodo no segundo circuito



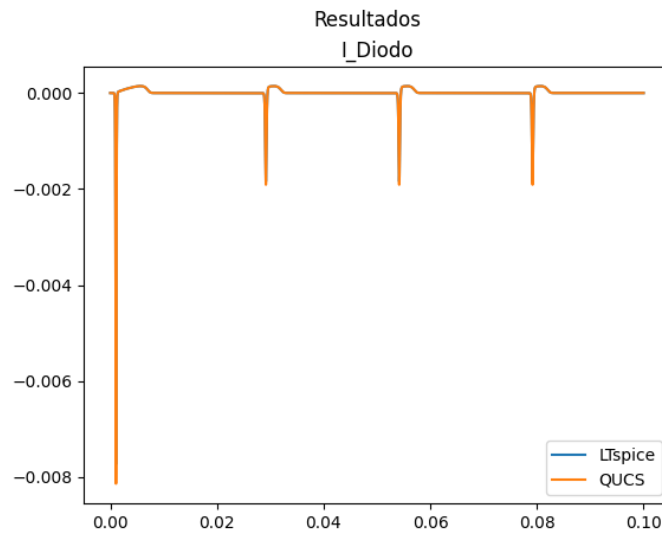
Fonte: Produção do próprio autor.

Gráfico 8 – Erros obtidos para a corrente sobre a carga e capacitor no segundo circuito



Fonte: Produção do próprio autor.

Gráfico 9 – Erros obtidos para a corrente sobre o diodo no segundo circuito



Fonte: Produção do próprio autor.

Os valores das variáveis que mensuram o erro, que foram calculados em relação ao LTspice e o QUCS, encontram-se nas Tabelas 14 e 15.

Tabela 14 – Variáveis de erro calculadas em relação ao LTspice para o segundo circuito

Variável	V_Fonte (V)	V_Carga (V)	V_Diodo (V)	I_Carga_e_Capacitor (A)	I_Diodo (A)
EA _{máximo}	6.28e-03	1.28e-02	6.09e-03	7.73e-03	7.73e-03
EA _{médio}	4.00e-03	5.88e-03	3.15e-03	7.79e-05	4.43e-05
V _{absmax}	5.00e+00	3.80e+00	4.39e+00	1.22e-01	1.22e-01
% EA _{máximo}	0.126	0.337	0.139	6.34	6.34
% EA _{médio}	0.08	0.155	0.0718	0.0639	0.0364

Fonte: Produção do próprio autor.

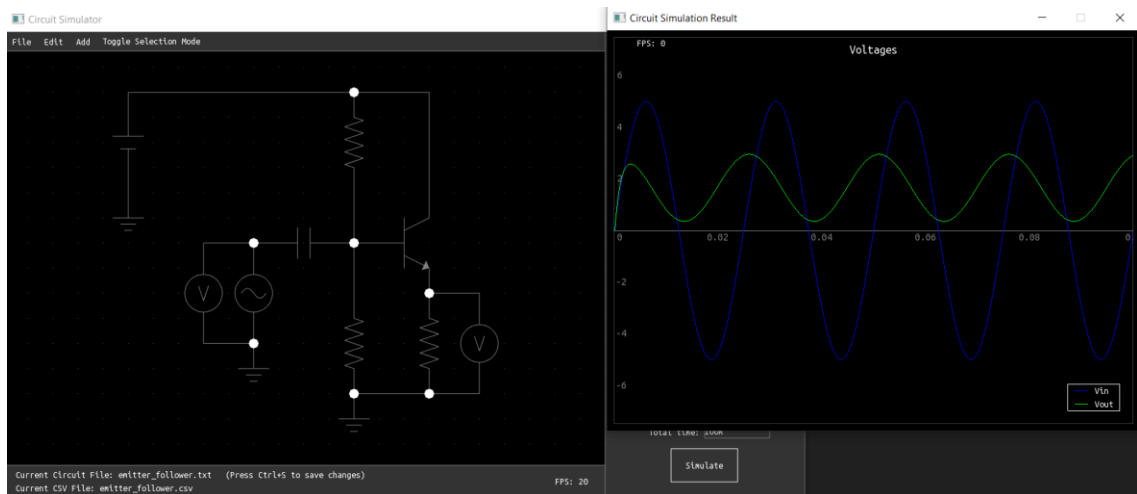
Tabela 15 – Variáveis de erro calculadas em relação ao QUCS para o segundo circuito

Variável	V_Fonte (V)	V_Carga (V)	V_Diodo (V)	I_Carga_e_Capacitor (A)	I_Diodo (A)
EA _{máximo}	6.28e-03	1.35e-02	6.41e-03	8.14e-03	8.14e-03
EA _{médio}	4.00e-03	6.43e-03	3.35e-03	7.99e-05	4.56e-05
V _{absmax}	5.00e+00	3.80e+00	4.39e+00	1.22e-01	1.22e-01
% EA _{máximo}	0.126	0.355	0.146	6.67	6.67
% EA _{médio}	0.08	0.169	0.0761	0.0656	0.0374

Fonte: Produção do próprio autor.

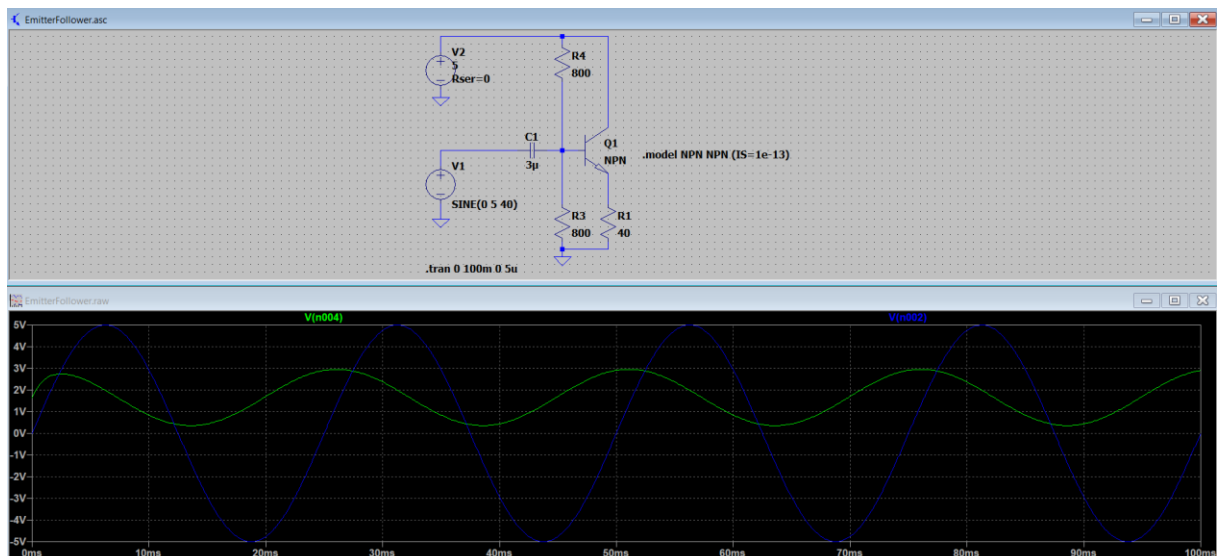
O circuito montado para a análise seguinte foi um seguidor de emissor, que consiste em duas fontes de tensão, uma contínua de 5 V e outra senoidal com amplitude de 5 V e frequência de 40 Hz; um capacitor de 3 μF ; três resistores, dois de 800 Ω e um de 40 Ω ; e um BJT do tipo NPN, com corrente de saturação igual a $1\text{e-}13$ A, a uma temperatura de 27 $^{\circ}\text{C}$ e demais parâmetros iguais ao modelo padrão do SPICE. Os resultados estão apresentados nas Figuras 31, 32 e 33.

Figura 31 – Diagrama e resultados para o terceiro circuito no simulador desenvolvido



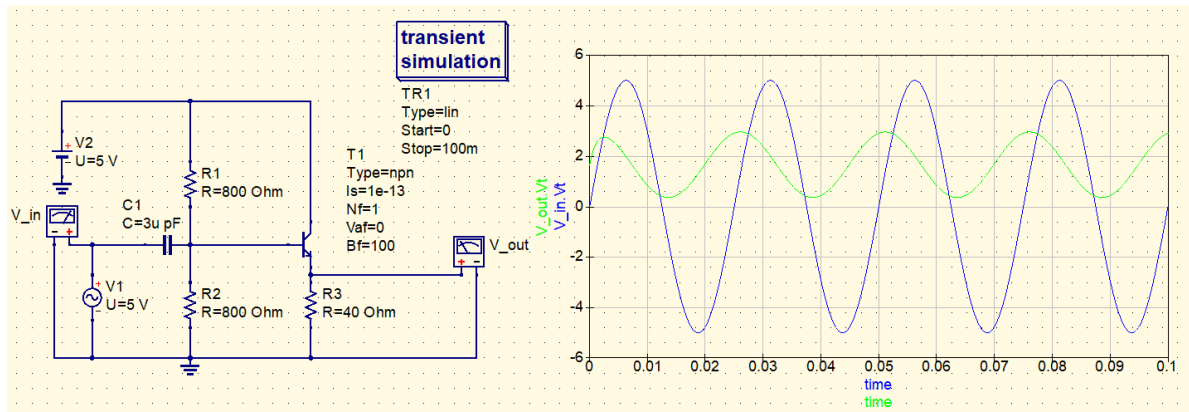
Fonte: Produção do próprio autor.

Figura 32 – Diagrama e resultados para o terceiro circuito no LTSpice



Fonte: Produção do próprio autor.

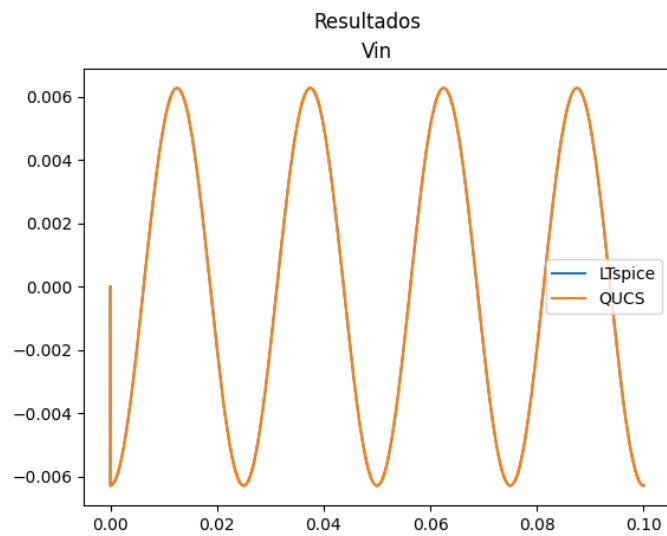
Figura 33 – Diagrama e resultados para o terceiro circuito no QUCS



Fonte: Produção do próprio autor.

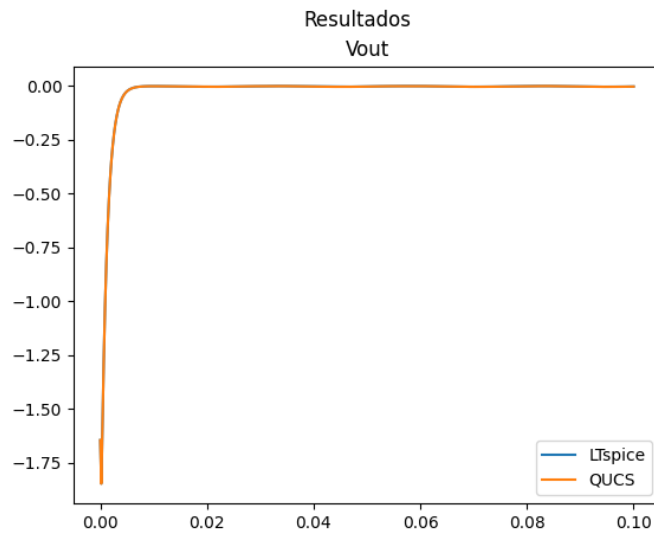
Os gráficos que quantificam a diferença entre os valores encontrados pelo simulador desenvolvido e os de referência encontram-se nos Gráficos 10 e 11.

Gráfico 10 – Erros obtidos para a tensão de entrada no terceiro circuito



Fonte: Produção do próprio autor.

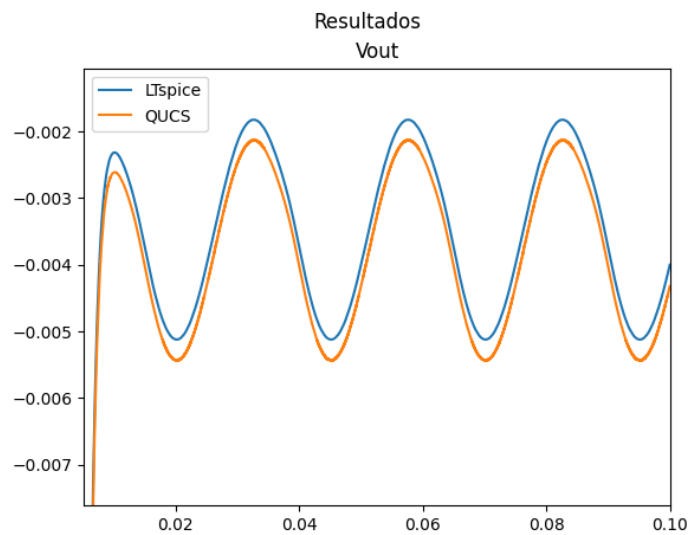
Gráfico 11 – Erros obtidos para a tensão de saída no terceiro circuito



Fonte: Produção do próprio autor.

Para esse circuito, notou-se uma diferença relativamente grande no valor da tensão de saída para os instantes iniciais da simulação. O Gráfico 12 possui uma versão amplificada do Gráfico 11, para o intervalo de tempo após a estabilização do erro em uma amplitude menor.

Gráfico 12 – Ampliação do gráfico dos erros obtidos para a tensão sobre a carga no terceiro circuito



Fonte: Produção do próprio autor.

Os resultados para dos cálculos das variáveis de erro encontram-se nas Tabelas 16 e 17.

Tabela 16 – Variáveis de erro calculadas para o circuito seguidor de emissor em relação ao LTspice

Variável	Vin (V)	Vout (V)
EA _{máximo}	6.28e-03	1.85e+00
EA _{médio}	4.00e-03	2.80e-02
V _{absmax}	5.00e+00	2.95e+00
% EA _{máximo}	0.126	62.5
% EA _{médio}	0.08	0.946

Fonte: Produção do próprio autor.

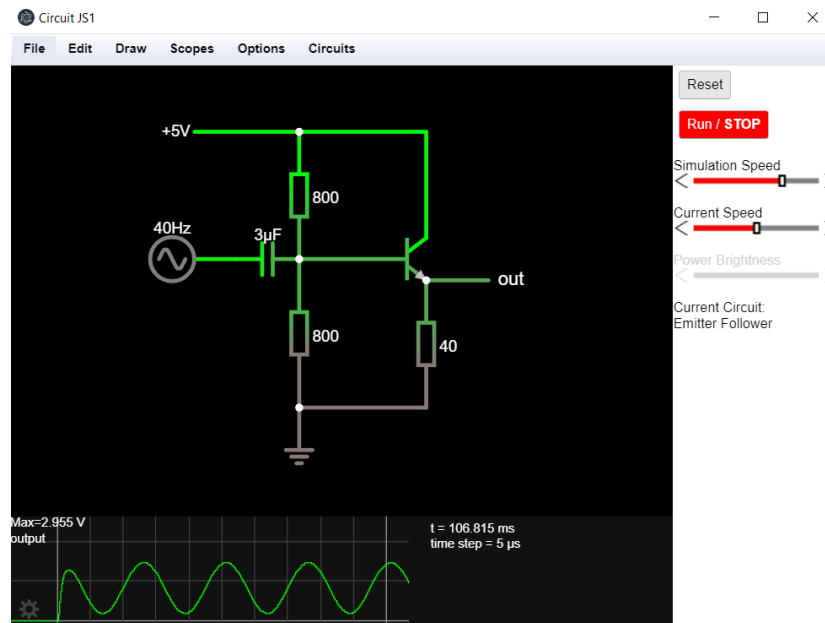
Tabela 17 – Variáveis de erro calculadas para o circuito seguidor de emissor em relação ao QUCS

Variável	Vin (V)	Vout (V)
EA _{máximo}	6.28e-03	1.85e+00
EA _{médio}	4.00e-03	2.83e-02
V _{absmax}	5.00e+00	2.96e+00
% EA _{máximo}	0.126	62.5
% EA _{médio}	0.08	0.956

Fonte: Produção do próprio autor.

Embora o seguidor de emissor apresente um comportamento diferente no início da simulação em comparação ao LTspice e QUCS, os resultados são bastante próximos aos calculados pelo simulador *Falstad*. Isso pode ser constatado comparando as Figuras 34 e 31, que apresentam curvas similares nos instantes iniciais da simulação.

Figura 34 – Diagrama e resultados para o circuito seguidor de emissor no *Falstad Circuit Simulator*

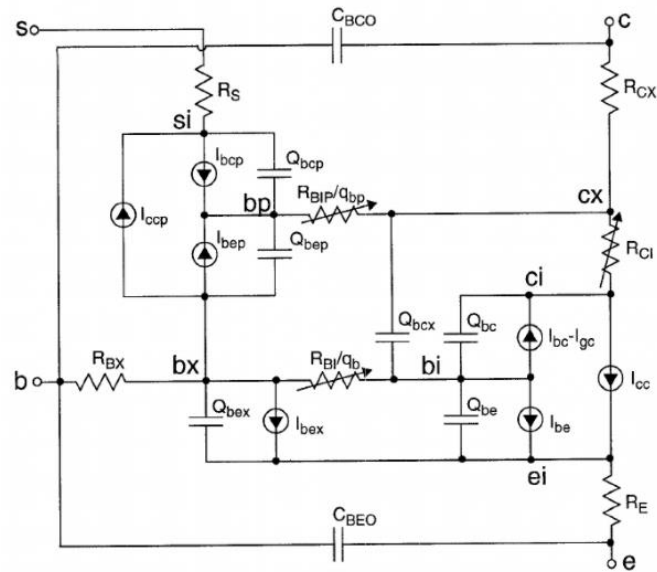


Fonte: Produção do próprio autor.

Não foi possível realizar um estudo ponto a ponto com relação às formas de onda geradas pelo simulador *Falstad*, pois o programa não oferece a opção de salvar os formatos de onda resultantes em arquivos CSV ou de texto, embora permita armazenar a configuração dos circuitos. Contudo, a análise de alguns pontos através da própria interface da aplicação comprovou a proximidade dos valores com precisão até a segunda casa decimal.

A partir da análise dos gráficos de erro e das variáveis que o quantificam, notou-se que os maiores desvios entre os resultados obtidos para os simuladores próprio e de referência ocorrem no início dos gráficos, especialmente para circuitos não lineares. Uma das possíveis causas para essas diferenças pode ser a relativa simplicidade dos modelos implementados para o diodo e transistor bipolar de junção, que não considera todos os parâmetros que os simuladores de referência utilizam. A Figura 35 mostra o modelo do BJT usado pelo LTspice, que utiliza dezenas de fatores que agregam efeitos de temperatura, frequência de operação, ruído e outras características construtivas do elemento. O simulador QUCS utiliza modelos similares.

Figura 35 – Modelo do transistor BJT NPN usado pelo LTspice



Fonte: McAndrew *et al.* (1995).

As diferenças entre os modelos das Figuras 8 e 35 são comparáveis às encontradas para outros componentes não lineares (diodo e MOSFET).

O uso de intervalos de tempo adaptativos também pode proporcionar diferenças significativas entre os resultados, especialmente instantes de tempo onde ocorrem transições rápidas de tensão/corrente. Isso pode ser particularmente observado nos gráficos para o retificador de onda completa com filtro capacitivo, onde os maiores erros ocorrem nos momentos em que os diodos mudam seu modo de operação de condução para corte e vice-versa.

O ponto de operação inicial do circuito também pode ter grandes variações em função das condições iniciais assumidas para os elementos de cada modelo. Com a influência das fontes independentes do circuito, as variáveis de medição de ambos os modelos gradualmente convergem para valores próximos, reduzindo o erro, como pode ser observado no Gráfico 11.

Outra provável razão para as diferenças observadas é a implementação do método de Newton-Raphson, que é realizada de forma mais rigorosa pelos simuladores de referência. Nesse caso, um fator importante é a aplicação de condições de convergência que envolvem a corrente nos ramos do circuito, o que não foi feito no simulador desenvolvido, por fins de simplicidade. O uso de restrições mais brandas nessa etapa do ciclo de simulação pode acarretar a acumulação

de eventuais erros de cálculo, que podem aumentar ou diminuir dependendo da estabilidade do circuito para as condições impostas. Esse fato pode ser comprovado pela proximidade da simulação do seguidor de emissor feita pelo *Falstad*, que também apenas considera a tensão para determinar a convergência do método de Newton-Raphson.

Em menor escala, outras possíveis fontes de erro estão relacionadas a:

- O processo de exportação dos resultados para CSV: os programas de referência têm um limite no número de casas decimais usadas para expressar os valores nesse tipo de arquivo (6 casas para o LTspice; 5 para o QUCS);
- Erros de truncamento e arredondamento decorrentes de operações com números de ponto flutuante.
- Para o caso do simulador LTspice, a distribuição dos pontos simulados para os circuitos não é homogênea, como foi discutido anteriormente. O processo de interpolação linear efetuado para simplificar as comparações realizadas também pode gerar imprecisões nos resultados.

De forma geral, o simulador desenvolvido apresentou um desempenho razoável no quesito precisão, apresentando um erro absoluto máximo de 6,67% para os dois primeiros testes realizados. No terceiro circuito, o maior erro percentual (62,5%) ocorreu no início da simulação, porém convergiu rapidamente para valores menores, permanecendo em uma faixa relativamente próxima de zero até o fim da simulação.

Em contraste, o erro médio absoluto máximo encontrado dentre os três circuitos foi de 0,956%, indicando que as maiores imprecisões geralmente ocorrem em momentos específicos da simulação, como mudanças abruptas de tensão e corrente, enquanto os valores são mais precisos em outras situações.

As tabelas também mostram que, para circuitos puramente lineares, a precisão é maior, pois várias das fontes de erros citadas não se aplicam. Este fato pode ser demonstrado pelos erros absoluto máximo e médio absoluto máximo encontrados para o circuito RLC série, que foram iguais a 0,13% e 0,0407%, respectivamente.

4.2.2 Usabilidade do simulador

Segundo a Associação Brasileira de Normas Técnicas (2002, p.3), a usabilidade é definida como a “medida na qual um produto pode ser usado por usuários específicos para alcançar objetivos específicos com eficácia, eficiência e satisfação em um contexto específico de uso”. Nesse contexto, a eficácia pode ser entendida como a acurácia e integralidade nas quais os objetivos específicos são atingidos; a eficiência, como a relação entre os recursos utilizados o grau de alcance dos objetivos; e a satisfação, como a ausência de desconforto ao utilizar um produto.

A satisfação ao utilizar um sistema é uma característica parcialmente subjetiva. O conforto ao utilizar uma nova ferramenta por um usuário depende, em grande parte, de fatores como experiências anteriores com programas similares e preferências pessoais envolvendo o posicionamento e atalho para o acesso a certos tipos de funções (*layout*). Entretanto, existem algumas características gerais que podem facilitar a adaptação do usuário, como a disposição de campos e a simplicidade na execução de tarefas. Esses itens foram analisados para o simulador do projeto e para duas outras ferramentas: LTspice, QUCS.

O simulador LTspice também apresenta uma interface gráfica relativamente simples, porém com funcionamento mais complexo que o simulador deste projeto, por possuir mais recursos. Uma de suas grandes vantagens é possibilidade de customização de atalhos, posicionamento de painéis e aparência. Além disso, por padrão, o simulador conta com uma grande gama de modelos para diferentes componentes, que utilizam modelos personalizados do SPICE e também são compatíveis com outras aplicações. Para mover elementos do circuito, é necessário que o programa seja colocado em um modo específico, o que pode ser pouco intuitivo para alguns usuários. Para a medição de tensão e corrente, o LTspice não apresenta voltímetros e amperímetros, sendo necessário que o usuário selecione com mouse o nó cuja tensão deve ser medida, ou arraste uma ponta de prova do primeiro ao segundo nó, caso o último não seja a referência. No caso de correntes, deve-se selecionar o componente cuja corrente deve ser analisada. Para novos usuários, esse método de seleção de variáveis de leitura pode ser pouco familiar, em comparação ao uso de medidores no diagrama.

O QUCS é outra aplicação com uma interface limpa, mas com várias funcionalidades interessantes. O programa conta com uma barra lateral usada para a adição de componentes, medidores, gráficos e parâmetros de simulação, sendo que esses elementos podem ser colocados na mesma área. Isso traz como vantagem a possibilidade de condensar várias informações importantes da simulação em uma única região, o que facilita a rápida análise pelo usuário. Esse simulador também possibilita o movimento de objetos de forma intuitiva, através do arraste do mouse (*click and drag*), além de facilitar a rotação e espelhamento dos mesmos através de botões na interface.

A interface do simulador apresentado nesse projeto foi desenvolvida com base nas duas ferramentas analisadas e em outras, visando amenizar algumas das desvantagens já apontadas, e reutilizar algumas das vantagens apresentadas para os simuladores de referência. Para isso, optou-se por uma interface minimalista majoritariamente baseada em escala de cinza, que, embora não seja visualmente apelativa, apresenta os parâmetros de simulação de forma clara e concisa através de uma barra lateral. Componentes selecionados apresentam uma cor distinta (verde), sendo que seus respectivos parâmetros também podem ser consultados através da barra à direita. Na porção inferior da janela, informações relacionadas aos arquivos de texto e CSV, nos quais os dados dos circuitos montados são salvos, são sempre mostrados ao usuário. A navegação para a escolha de funções/elementos para o circuito pode ser realizada inteiramente por meio do mouse através da barra de menus. Algumas opções, como a adição de componentes básicos (resistores, capacitores, fontes, etc.), edição (copiar, colar, cortar), e armazenamento de diagrama, também possuem atalhos no teclado que proporcionam sua execução direta pelo usuário. Funções de cópia, recorte e colagem de componentes são usadas de modo parecido com aquele presente em editores de texto, facilitando sua utilização para grande parte dos usuários, já que essa forma de aplicação é bastante comum. Operações que envolvem o movimento e rotação de componentes podem ser realizadas no mesmo modo de seleção, através do arraste do mouse ou da seleção dos botões, respectivamente. Opções de *zoom* e *pan* são feitas por meio do botão *scroll* (meio) do mouse, de forma similar à maioria dos programas de montagem não só de circuitos elétricos, mas de desenho em geral.

Uma possível forma de avaliar a usabilidade da interface na prática seria a condução de testes com usuários, que posteriormente comparariam a aplicação com outros simuladores. Contudo, devido às restrições impostas pela pandemia, não foi possível realizar esse procedimento, por

conta das dificuldades técnicas envolvidas. Um fator importante seria a disponibilidade dos entrevistados, que teriam que preparar seus computadores pessoais para a realização do experimento. Idealmente, o teste e a posterior coleta de dados deveriam ser realizados em um espaço próprio, onde o entrevistador prepararia os equipamentos antecipadamente, o que reduziria o tempo necessário para a realização do teste, além de prevenir eventuais problemas durante sua execução.

5 CONCLUSÕES

Os primeiros simuladores apresentavam interfaces gráficas pouco otimizadas para a experiência do usuário, tornando-os ferramentas destinadas principalmente a indivíduos com o devido conhecimento técnico da aplicação associada. Contudo, houve uma evolução nas tecnologias de desenvolvimento de *software* nas últimas décadas, que possibilitou a concepção de ferramentas de simulação cada vez mais acessíveis ao público em geral, aliadas a um desempenho robusto e eficiente. Atualmente, existem vários programas especializados nas demandas das diversas áreas da engenharia elétrica, como eletrônica de potência, sistemas digitais, sistemas de energia e engenharia de radiofrequência. Outra aplicação importante, sendo inclusive o foco deste projeto, foi a educacional, que, por ser direcionada a estudantes e professores, prioriza a usabilidade e simplicidade.

Considerando que o código do simulador desenvolvido não utiliza nenhuma biblioteca externa para a implementação dos algoritmos computacionais relacionados ao fluxograma básico de uma simulação transiente do SPICE, a semelhança de seus resultados com aqueles previstos por aplicações já consolidadas no mercado permite concluir que o programa obteve um desempenho satisfatório com relação a sua precisão. Além disso, o trabalho pode atuar como uma ferramenta de apoio ao ensino, visto que apresenta um estudo detalhado do funcionamento de simuladores de circuitos elétricos, incluindo seus aspectos teóricos.

Com relação à sua interface gráfica, buscou-se produzir uma aplicação simples, que pudesse combinar as vantagens de alguns simuladores já existentes, aliados à tentativa de melhorar o funcionamento de alguns mecanismos que podem trazer desconforto a usuários pouco experientes.

Como parte de trabalhos futuros, existem vários recursos que podem ser agregados ao simulador. Para torná-lo mais preciso com circuitos não lineares, é possível melhorar a aplicação do método de Newton-Raphson, incluindo o comportamento das correntes nas condições de convergência. A precisão geral do programa também pode ser melhorada com a introdução do cálculo de intervalos de integração adaptativos, o que ainda ajudaria com problemas de convergência. Além disso, pode-se aprimorar os modelos dos componentes, considerando efeitos como temperatura e frequência de operação, oferecendo, inclusive, a

possibilidade de escolher o tipo de modelo, como é feito pelos simuladores LTspice e QUCS. Outra funcionalidade interessante seria a criação de blocos que abstraem circuitos complexos em um único componente, de modo que apenas suas entradas e saídas fossem acessíveis. Essa alteração possibilitaria a criação de modelos pelo próprio usuário, aumentando o escopo do simulador.

Para a interface, uma adição interessante seria a possibilidade de mostrar os nomes dos componentes próximos a sua representação em diagrama, para facilitar sua identificação. Também existe potencial para a melhora dos menus de adição de componentes e armazenamento de arquivos, que poderiam apresentar mais opções. Outra capacidade comum em simuladores é a medição da potência instantânea de componentes do circuito. Isto poderia ser incluído ao projeto por meio da adição de um wattímetro, que internamente operaria como a junção de um voltímetro e amperímetro, tendo como saída a multiplicação de suas leituras instantâneas. Em geral, a customização da aparência, controles e atalhos também tornaria o programa mais agradável ao usuário com vários níveis de experiência prévia.

As crescentes vantagens trazidas por ferramentas de simulação têm tornado seu uso cada vez mais comum em diversas áreas da engenharia. Embora os algoritmos e modelos aplicados estejam cada vez mais avançados, o estudo do seu funcionamento é imprescindível, pois proporciona um melhor entendimento de suas possibilidades e limitações. Por fim, simuladores propiciam grandes benefícios no contexto educacional, por serem ferramentas que complementam o aprendizado teórico.

REFERÊNCIAS BIBLIOGRÁFICAS

- ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR 9241-11**: Requisitos Ergonômicos para Trabalho de Escritórios com Computadores: Parte 11 – Orientações sobre Usabilidade. Rio de Janeiro: ABNT, 2002.
- BLUME, W. Computer circuit simulation: Circuit modelling programs make breadboards obsolete. **Byte**, [s. l.], v. 11, n.7, p. 165-170, 1986. Disponível em: <https://www.cl.cam.ac.uk/teaching/1314/NumMethods/supporting/spice-wolfram-blume-byte1986.pdf>. Acesso em: 10 abr. 2021.
- BURKS, A. W.; BURKS, A. R. First General-Purpose Electronic Computer. **IEEE Annals of the History of Computing**, [s. l.], v. 3, n. 4, p. 310-389, out./dez. 1981.
- FALSTAD, P. **pfalstad/circuitjs1**: Electronic Circuit Simulator in the Browser. [S. l.], 2021. Disponível em: <https://github.com/pfalstad/circuitjs1>. Acesso em: 6 set. 2021.
- FOURMENT, M.; GILLINGS, M. R. A comparison of common programming languages used in bioinformatics. **BMC Bioinformatics**, [s. l.], v. 9, feb. 2008. Disponível em: <https://bmcbioinformatics.biomedcentral.com/track/pdf/10.1186/1471-2105-9-82.pdf>. Acesso em: 7 ago. 2021.
- GOLDSMAN, D.; NANCE, R. E. A brief history of simulation. *In*: WINTER SIMULATION CONFERENCE, 2009, Austin. **Proceedings** [...]. [S. l.]: IEEE, 2009. p. 310-313. Disponível em: <https://www.informs-sim.org/wsc09papers/028.pdf>. Acesso em: 21 mar. 2021.
- GOMILA, L. **Simple and Fast Multimedia Library**, 2021. Página inicial. Disponível em: <https://www.sfml-dev.org/index.php>. Acesso em: 1 ago. 2021.
- HERBST, Steven; LEVITT, Antoine. **Companion Models for Basic Non-Linear and Transient Devices**. [S. l.: s. n.], 2008. 14 p. Disponível em: <http://dev.hypertriton.com/edacious/trunk/doc/lec.pdf>. Acesso em: 6 ago. 2021.
- HYNDMAN, R. J.; ATHANASOPOULOS, G. **Forecasting: principles and practice**. 2. ed. Melbourne: OTexts, 2018. Disponível em: <https://otexts.com/fpp2/index.html>. Acesso em: 6. set. 2021.
- JAHN, S.; MARGRAF, M.; HABCHI, V.; JACOB, R. **QUCS: Technical Papers**. [S. l.: s. n.], 2007. 263 p. Disponível em: <http://qucs.sourceforge.net/docs/technical/technical.pdf>. Acesso em: 19 set. 2021.
- KUNDERT, K. S.; CLIFFORD, I. H. Achieving accurate results with a circuit simulator. *In*: IEE Colloquium on SPICE: Surviving Problems in Circuit Evaluation, 1993, London. **Proceedings** [...]. [S. l.: s. n.], 1993. Disponível em: <https://kenkundert.com/docs/eda+t93-paper.pdf>. Acesso em: 14 ago. 2021.

LANNUTTI, F.; NENZI, P.; OLIVIERI, M. KLU sparse direct linear solver implementation into NGSPICE. *In: INTERNATIONAL CONFERENCE MIXED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS*, 19., 2012, Varsóvia. **Proceedings** [...]. [S. l.]: IEEE, 2012. p. 69-73. Disponível em:

https://www.researchgate.net/publication/254041555_KLU_sparse_direct_linear_solver_implementation_into_NGSPICE. Acesso em: 8 jul. 2021.

LAY, D. C.; LAY, S. R.; MCDONALD, J. J. **Linear Algebra and Its Applications**. 5. ed. New York: Pearson, 2015.

LITOVSKI, V.; ZWOLINSKI, M. **VLSI Circuit Simulation and Optimization**. 1. ed. Londres: Chapman and Hall, 1997.

MAUCHLY, J. W. The use of High Speed Vacuum Tube Devices for Calculating. *In: RANDEL, B. The Origins of Digital Computers*. 3. ed. New York: Springer-Verlag, 1982. p. 355-358.

MCANDREW, C.; SEITCHIK J.; BOWERS, D.; DUNN, M.; FOISY, M.; GETREU, I.; MCSWAIN, M.; MOINIAN, S.; PARKER, J.; WIJNEN, P.; WAGNER, L. Vertical Bipolar Inter Company 1995: An Improved Vertical, IC Bipolar Transistor Model. *In: BIPOLAR/BICMOS CIRCUITS AND TECHNOLOGY MEETING*, 1995, Minneapolis. **Proceedings** [...]. [S. l.]: IEEE, 1996, p. 170-177. Disponível em: https://www.researchgate.net/publication/3627075_VBIC95_An_improved_vertical_IC_bipolar_transistor_model. Acesso em: 4 set. 2021.

MCNEILL, D. Analog circuit analysis: An analog circuit modeling and simulation program for the Commodore 64. **Byte**, [s. l.], v. 11, n.7, p. 170-178, 1986. Disponível em: <https://www.cl.cam.ac.uk/teaching/1314/NumMethods/supporting/spice-wolfram-blume-byte1986.pdf>. Acesso em: 10 abr. 2021.

NAGEL, L. W.; PEDERSON, D. O. SPICE (Simulation Program with Integrated Circuit Emphasis). *In: MIDWEST SYMPOSIUM ON CIRCUIT THEORY*, 16., 1973, Waterloo. **Proceedings** [...]. [S. l.: s. n.], 1973. Disponível em: <https://www2.eecs.berkeley.edu/Pubs/TechRpts/1973/ERL-382.pdf>. Acesso em: 28 mar. 2021.

NEWTON, A. R.; PEDERSON, D. O.; SANGIOVANNI-VINCENTELLI, A. **SPICE3 Version 3f3 User's Manual**. [S. l.: s. n.], 1993. Disponível em: http://fides.fe.uni-lj.si/spice/download/spice3_manual.pdf. Acesso em: 3 abr. 2021.

NGADA, N. M. Simulations. *In: CAS - CERN ACCELERATOR SCHOOL: POWER CONVERTERS*, 2014, Baden. **Proceedings** [...]. Genebra: Ed. CERN, 2015. p. 395-414. Disponível em: <https://cds.cern.ch/record/2038679/files/395-414-Ngada.pdf>. Acesso em: 29 mar. 2021.

NILSSON, J. W.; RIEDEL, S. A. **Electric Circuits**. 9. ed. Upper Saddle River: Prentice Hall, 2011.

NORDGREN, W. B. Flexsim simulation environment. *In: WINTER SIMULATION CONFERENCE*, 2002, San Diego. **Proceedings** [...]. [S. l.]: IEEE, 2003. Disponível em: <https://ieeexplore.ieee.org/document/1172892?denied=>. Acesso em: 10 jul. 2021.

PEDERSON, D. O. A Historical Review of Circuit Simulation. **IEEE Transactions on Circuits and Systems**, [s. l.], v. 31, n. 1, p. 103-111, jan. 1984. Disponível em: <http://web.engr.oregonstate.edu/~karti/ece521/dop.pdf>. Acesso em: 21 mar. 2021.
PILLAGE, L. T.; ROHRER, R. A.; VISWESWARIAH, C. **Electronic Circuit and System Simulation Methods**. [S. l.]: McGraw-Hill, 1994.

PRESS, W. H.; TEUKOLSKY, S. A.; VETTERLING, W. T.; FLANNERY, B. P. **Numerical Recipes: The Art of Scientific Computing**. 3. ed. New York: Cambridge University Press, 2007.

PRODANOV, C. C.; FREITAS, E. C. **Metodologia do Trabalho Científico: Métodos e Técnicas da Pesquisa e do Trabalho Acadêmico**. 2. ed. Novo Hamburgo: Ed. Universidade Feevale, 2013.

QUCS TEAM. Qucs project: F. A. Q. *In: Frequently asked questions*. [S. l.], 2017. Disponível em: <http://qucs.sourceforge.net/faq.html>. Acesso em: 4 set. 2021.

QUEIROZ, A. C. M. Compact nodal analysis with controlled sources modeled by ideal operational amplifiers. *In: MIDWEST SYMPOSIUM ON CIRCUITS AND SYSTEMS*, 38., 1995, Rio de Janeiro. **Proceedings** [...]. [S. l.: s. n.], 1995. Disponível em: <https://www.coe.ufrj.br/~acmq/papers/compact1995.pdf>. Acesso em 16 jul. 2021.

ROSEN, S. **The Origins of Modern Computing**. West Lafayette: Ed. Universidade Purdue, 1990. Disponível em: <https://docs.lib.purdue.edu/cgi/viewcontent.cgi?article=1014&context=cstech>. Acesso em: 20 mar. 2021.

SHANNON, R. E. Introduction to the art and science of simulation. *In: WINTER SIMULATION CONFERENCE*, 1998, Washington. **Proceedings** [...]. [S. l.]: IEEE, 1998. p. 7-14. Disponível em: <https://www.informs-sim.org/wsc98papers/001.PDF>. Acesso em: 20 mar. 2021.

SIMULAÇÃO. *In: DICIONÁRIO de língua portuguesa*. [S. l.]: Melhoramentos, 2021. Disponível em: <https://michaelis.uol.com.br/moderno-portugues/busca/portugues-brasileiro/simulacao>. Acesso em: 20 mar. 2021.

SPINOLA, D. S. **Estrutura produtiva no modelo Neokaleckiano de crescimento e distribuição: simulações para a economia brasileira**. 2014. Dissertação (Mestrado em Ciências Econômicas) – Universidade Estadual de Campinas, Instituto de Economia, Campinas, 2014. Disponível em: <http://www.repositorio.unicamp.br/handle/REPOSIP/286498>. Acesso em: 10 jul 2021.

TUMA, T.; BUERMEN, Á. **Circuit Simulation with SPICE OPUS: Theory and Practice**. [S. l.]: Birkhäuser, 2009.