



**UNIVERSIDADE ESTADUAL DA PARAÍBA  
CAMPUS VII – ANTONIO MARIZ  
CENTRO DE CIÊNCIAS EXATAS E SOCIAIS APLICADAS  
CURSO DE LICENCIATURA PLENA EM FÍSICA**

**JERFFESON RAMON GUIMARÃES LOPES**

**O MÉTODO NUMÉRICO DAS DIFERENÇAS FINITAS APLICADO À  
PROBLEMAS DE SEGUNDA ORDEM COM APLICAÇÃO NA ANÁLISE E  
MODELAGEM DE CIRCUITOS RC/RLC**

**PATOS  
2018**

**JERFFESON RAMON GUIMARÃES LOPES**

**O MÉTODO NUMÉRICO DAS DIFERENÇAS FINITAS APLICADO À  
PROBLEMAS DE SEGUNDA ORDEM COM APLICAÇÃO NA ANÁLISE E  
MODELAGEM DE CIRCUITOS RC/RLC**

Trabalho de Conclusão de Curso apresentado ao Curso de Licenciatura Plena em Física da Universidade Estadual da Paraíba, em cumprimento à exigência para obtenção do grau de Licenciado em Física.

**Área de concentração:** Física computacional.

**Orientador:** Prof. Dr. Rodrigo César Fonseca da Silva.

**PATOS  
2018**

É expressamente proibido a comercialização deste documento, tanto na forma impressa como eletrônica. Sua reprodução total ou parcial é permitida exclusivamente para fins acadêmicos e científicos, desde que na reprodução figure a identificação do autor, título, instituição e ano do trabalho.

L864m Lopes, Jerffeson Ramon Guimarães.

O método numérico das diferenças finitas aplicado à problemas de segunda ordem com aplicação na análise e modelagem de circuitos RC/RLC [manuscrito] / Jerffeson Ramon Guimaraes Lopes. - 2018.

66 p. : il. colorido.

Digitado.

Trabalho de Conclusão de Curso (Graduação em Física) - Universidade Estadual da Paraíba, Centro de Ciências Exatas e Sociais Aplicadas , 2018.

"Orientação : Prof. Dr. Rodrigo César Fonseca da Silva ,  
Coordenação do Curso de Física - CCEA."

1. Métodos numéricos. 2. Computação numérica. 3.  
Diferenças finitas. 4. Circuitos elétricos. I. Título

21. ed. CDD 512

JERFFESON RAMON GUIMARÃES LOPES

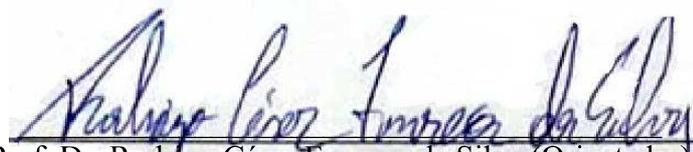
O MÉTODO NUMÉRICO DAS DIFERENÇAS FINITAS APLICADO À PROBLEMAS DE  
SEGUNDA ORDEM COM APLICAÇÃO NA ANÁLISE E MODELAGEM DE  
CIRCUITOS RC/RLC

Trabalho de Conclusão de Curso apresentado  
ao Curso de Licenciatura Plena em Física da  
Universidade Estadual da Paraíba, em  
cumprimento à exigência para obtenção do  
grau de Licenciado em Física.

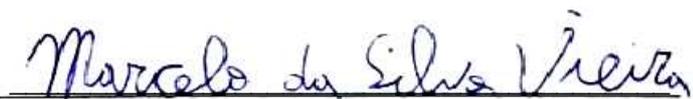
Área de concentração: Física computacional.

Aprovada em: 06/12/2018.

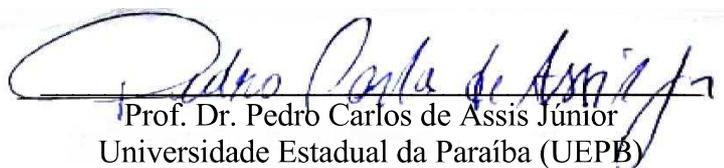
**BANCA EXAMINADORA**



Prof. Dr. Rodrigo César Fonseca da Silva (Orientador)  
Universidade Estadual da Paraíba (UEPB)



Prof. Dr. Marcelo Vieira da Silva  
Universidade Estadual da Paraíba (UEPB)



Prof. Dr. Pedro Carlos de Assis Júnior  
Universidade Estadual da Paraíba (UEPB)

À minha Mãe e meu Pai, pela dedicação e amor que sempre tiveram comigo, à minha noiva Gessica, pelo amor e companheirismo, DEDICO.

## **AGRADECIMENTOS**

À Deus, pelo dom da vida, pois sem sua misericórdia e sua proteção divina nada sou. À Mãe Rainha, pois sei que sempre está me cobrindo com seu manto a todo o momento.

À minha mãe Sonia e meu pai Pedro Neto, por todo apoio, amor e carinho, que até os dias de hoje é de tamanho incomensurável.

À minha futura esposa Gessica, por todos os momentos maravilhosos que passamos juntos, pelo amor e carinho, que nos fortalecem a cada dia.

Ao meu Orientador, Prof. Dr. Rodrigo César Fonseca da Silva, que contribuiu e ajudou bastante, pois sem sua dedicação e orientação esse trabalho não teria acontecido.

Aos amigos Robson Amauri e Emerson Paes, que são dois irmãos que a graduação me concedeu.

Aos meus amigos, Eliton, Ademar, Fellipe e Almir, que ao longo dessa caminhada sempre me ajudaram.

Aos professores do Curso de Licenciatura Plena em Física da UEPB, em especial, Everton Cavalcante, Marcelo Vieira, Pedro Carlos e Kilmara Rodrigues, que contribuíram ao longo de 60 meses, por meio das disciplinas e debates, para o desenvolvimento deste trabalho.

Aos colegas de classe pelos momentos de amizade e apoio.

“Não importa o que você olha, se você olhar de perto o suficiente, você está envolvido em todo o universo.”

Michael Faraday

## RESUMO

O avanço computacional e a crescente necessidade de se obter soluções de problemas cada vez mais complexos permitiram o surgimento dos métodos numéricos. A ideia básica desses métodos é o processo de discretização, que reduz um problema contínuo, com número infinito de variáveis, em um problema discreto com um número finito de variáveis, podendo ser resolvido computacionalmente. Existem diversos tipos de métodos numéricos utilizados para encontrar a solução de uma equação diferencial, como: método dos elementos finitos, método dos elementos de contorno, método dos volumes finitos e o método das diferenças finitas. Nesse trabalho vamos operar a simulação com o Scilab, que é um software científico para computação numérica, utilizando o método das diferenças finitas com o objetivo de solucionar problemas que envolvem Equações Diferenciais de Segunda Ordem com aplicação direcionada à análise e simulação do comportamento dos circuitos elétricos RC (carregamento e descarregamento do capacitor) e RLC (paralelo e em série) no decorrer do tempo. Também realizaremos uma comparação entre a solução exata e a solução numérica obtida pelo método das diferenças finitas, com a finalidade de verificar a eficiência do método utilizado.

**Palavras-Chave:** Computação Numérica, Diferenças finitas, Scilab, Circuitos Elétricos.

## ABSTRACT

The computational advance and the increasing need to obtain solutions of increasingly complex problems allowed the emergence of numerical methods. The basic idea of these methods is the discretization process, which reduces a continuous problem with infinite number of variables in a discrete problem with a finite number of variables and can be solved computationally. There are several types of numerical methods used to find the solution of a differential equation, such as: finite element method, boundary element method, finite volume method and finite difference method. In this work we will operate the simulation with the Scilab, which is a scientific software for numerical computation, using the method of finite differences with the objective of solving problems involving Equations Second Order Differences with application directed to the analysis and simulation of the behavior of the electric circuits RC (capacitor loading and unloading) and RLC (parallel and serial) over time. We will also make a comparison between the exact solution and the numerical solution obtained by the finite difference method, in order to verify the efficiency of the method used.

**Keywords:** Numerical Computation, Finite Differences, Scilab, Electrical Circuits.

## LISTA DE ILUSTRAÇÕES

Figura 1	O primeiro computador digital eletrônico: ENIAC.....	14
Figura 2	Interface do Scilab: Prompt de comando.....	17
Figura 3	Exemplo do uso do comando “clear”, “clc”, “xgrid”, “xtitle”, “legend” e função “plot” (usando SCINOTES).....	20
Figura 4	Exemplo de mudança do domínio contínuo de coordenadas (x,y) para o discreto de coordenadas (i,j).....	21
Figura 5	Visualização dos limites laterais para uma determinada função contínua $u(x)$ .....	22
Figura 6	Circuito RC ligado a uma bateria. ....	25
Figura 7	Circuito RC ligado com a retirada da bateria. ....	27
Figura 8	Gráfico do carregamento/descarregamento do capacitor.....	28
Figura 9	Gráfico do carregamento/descarregamento do capacitor, utilizando o método de diferenças finitas.....	30
Figura 10	Circuito RLC paralelo (sem fonte).....	32
Figura 11	Gráfico do circuito RLC Paralelo (Caso Superamortecido).....	34
Figura 12	Gráfico do circuito RLC Paralelo (Caso Subamortecido).....	35
Figura 13	Gráfico do circuito RLC Paralelo (Caso Criticamente amortecido).....	36
Figura 14	Circuito RLC série (sem fonte).....	36
Figura 15	Gráfico do circuito RLC Série (Caso Superamortecido).....	38
Figura 16	Gráfico do circuito RLC Série (Caso Subamortecido).....	39
Figura 17	Gráfico do circuito RLC Série (Caso Criticamente amortecido).....	40
Figura 18	Gráfico do circuito RLC Série (Caso Subamortecido) e RLC Paralelo (Caso Superamortecido).....	43
Figura 19	Gráfico do circuito RLC Série (Caso Subamortecido) e RLC Paralelo (Caso Subamortecido).....	44
Figura 20	Gráfico do circuito RLC Série (Caso Subamortecido) e RLC Paralelo (Caso Criticamente Amortecido).....	45

## LISTA DE ABREVIATURAS E SIGLAS

- CACSD - Computer Aided Control System Design.
- EDO - Equação Diferencial Ordinária.
- ENIAC - Electrical Numerical Integrator and Computer.
- ENPC - École Nationale des Ponts et Chaussées.
- INRIA - Instituto Francês de Pesquisa em Ciência da Computação e Controle.
- MDF - Método das Diferenças Finitas.
- RLC - Resistor, Indutor e Capacitor.

## LISTA DE SÍMBOLOS

A -	Ampére
C -	Coulomb
$\Omega$ -	Ohm
F -	Farad
V -	Volt
m -	mili
$\mu$ -	micro
n -	nano
p -	pico
$\frac{d}{dt}$ -	Derivada de primeira ordem
$\frac{d^2}{dt^2}$ -	Derivada de segunda ordem
$\varepsilon$ -	Fonte de energia
$\alpha$ -	Frequência de Neper
$\omega_0$ -	Frequência angular de ressonância
$\omega_d$ -	Frequência de oscilação

# SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO.....</b>	<b>13</b>
<b>2</b>	<b>FUNDAMENTAÇÃO TEÓRICA.....</b>	<b>16</b>
<b>2.1</b>	<b>Software Scilab: Histórico e algumas ferramentas básicas de programação.....</b>	<b>16</b>
2.1.1	Comandos Básicos do Scilab.....	18
<b>2.2</b>	<b>Método das diferenças finitas.....</b>	<b>20</b>
2.2.1	Derivadas de 1ª ordem.....	21
2.2.1.1	Diferença progressiva.....	22
2.2.1.2	Diferença regressiva.....	23
2.2.1.3	Diferença central.....	23
2.2.2	Derivadas de 2ª ordem.....	24
<b>3</b>	<b>APLICAÇÃO: CIRCUITOS ELÉTRICOS RC E RLC.....</b>	<b>24</b>
<b>3.1</b>	<b>Dedução das equações de carga e descarga dos capacitores utilizando equações diferenciais de primeira ordem.....</b>	<b>24</b>
3.1.1	Circuito RC - Modelagem numérica com diferenças finitas .....	28
<b>3.2</b>	<b>Circuito RLC: Relação entre tensão e corrente.....</b>	<b>29</b>
3.2.1	Circuito RLC paralelo: Resposta natural.....	30
3.2.2	Circuito RLC série: Resposta natural.....	35
<b>3.3</b>	<b>Circuito RLC paralelo e série: Modelagem numérica.....</b>	<b>39</b>
3.3.1	Circuito RLC paralelo.....	39
3.3.2	Circuito RLC série.....	41
<b>4</b>	<b>CONCLUSÃO.....</b>	<b>45</b>
	<b>REFERÊNCIAS.....</b>	<b>47</b>
	<b>APÊNDICE A – (SIMULADOR) CÓDIGO DO PROGRAMA NO SCILAB.....</b>	<b>49</b>

## 1 INTRODUÇÃO

O avanço da ciência permite a criação de tecnologias cada vez mais inovadoras. No último século assistimos o surgimento das tecnologias eletrônica e digital, que continuam evoluindo a passos largos em nossa atualidade. Esta evolução se manifesta com surgimento de novas ferramentas para o avanço do conhecimento científico, fechando um ciclo virtuoso de um digno crescimento.

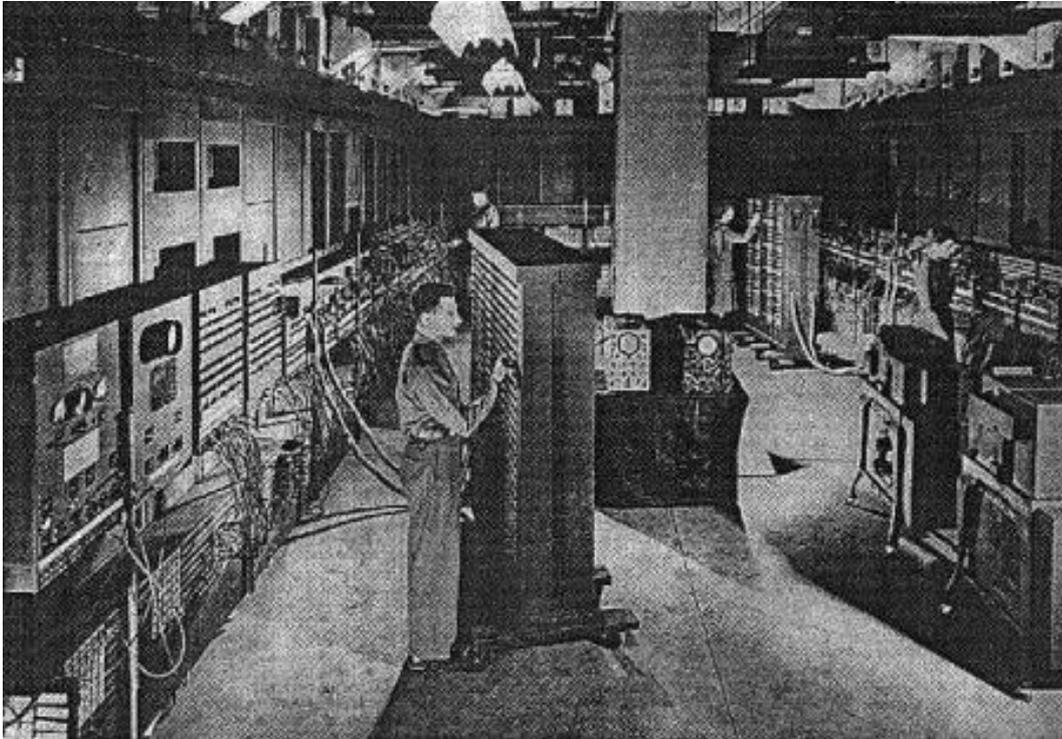
Com o passar dos anos, a Física tem ganhado um espaço cada vez mais amplo no cenário de aperfeiçoamento tecnológico, de modo que, os conceitos, leis e princípios físicos têm possibilitado a inovação e criação de produtos mais sofisticados, personalizados, seguros e eficazes. É através da inserção da Física na tecnologia que muitos produtos, como, por exemplo, a identificação por meio da íris ou mesmo as inovações dos sistemas de alarme utilizados nas residências, tem se disseminado e ganhado cada vez mais espaço no cotidiano.

A Física, como se sabe, é a ciência que estuda a natureza, buscando descrever e compreender os fenômenos que ocorrem nela. É por meio das compreensões, pesquisas, estudo dos princípios e leis que avanços tecnológicos têm surgido, dia a pós dia, com o intuito de modernizar processos e serviços e facilitar a vida das pessoas. Assim, o estudo da Física torna-se cada vez mais importante, pois, possibilita a compreensão dos conceitos e a aplicação dos mesmos na contínua evolução tecnológica.

Usando a Matemática como linguagem para “conversar” com a natureza, os primeiros pensadores que desenvolveram toda a ciência como conhecemos hoje, por meio de modelos que descrevem o comportamento do universo. As descrições exatas (analíticas) são muito raras em sistemas complexos, o que são os mais comuns. Kurt Gödel (1906-1978) constatou que não seria possível descrever toda a realidade matematicamente, uma vez que, qualquer sistema lógico não pode ser simultaneamente completo e consistente.

No ano de 1946, os Estados Unidos iniciaram o uso do primeiro computador eletrônico “ENIAC” (Electrical Numerical Integrator and Computer) criado em fevereiro de 1946, pelos cientistas norte-americanos John Eckert e John Mauchly, da Electronic Control Company. Depois de alguns anos, este tipo de tecnologia permitiria a realização de vários cálculos por segundo, algo impensável para um ser humano.

Figura 1: O primeiro computador digital eletrônico: ENIAC (Electrical Numerical Integrator and Calculator).



Fonte: <http://trabalhofernando.blogspot.com/2009/03/eniac-1946-primeiro-computador-digital.html>

Mesmo que a teoria matemática não pudesse descrever toda a realidade, seria possível aprofundar o conhecimento sobre alguns setores particulares. Hoje em dia, alguns supercomputadores instalados em centros de pesquisa são capazes de realizar dezenas de petaflops (flops = operações de ponto flutuante por segundo), ou seja, quatrilhões de operações por segundo. Com essa velocidade é possível simular fenômenos subatômicos, meteorológicos e até galácticos. Para desenvolver este tipo de simulação são necessários conhecimentos de Física e de Computação, formando uma nova disciplina interdisciplinar conhecida como Física Computacional.

Física Computacional é uma área multidisciplinar que engloba Física, Matemática Aplicada e Ciência da Computação, com o objetivo de resolver problemas nas ciências da natureza, sobretudo na Física. De modo geral, usa-se a Física Computacional quando não se consegue resolver um problema analiticamente ou há muitos dados para serem processados. A Física Computacional é fundamental, por exemplo, para armazenamento, distribuição e análise dos dados experimentais captados pelo Grande Colisor de Hádrons (a maior e mais poderosa máquina já feita pelo homem). Um assunto tradicional da Física Computacional é o uso de métodos computacionais para a solução de problemas que já possuem uma teoria

quantitativa. Esse estudo é importante, pois muitos sistemas físicos são descritos por equações que não podem ser resolvidas por meio da álgebra. As técnicas desenvolvidas permitem a solução de várias classes de problemas em diversas áreas da Física. São utilizados métodos de integração numérica para a solução de equações diferenciais ordinárias, métodos estatísticos como o método de Monte Carlo (qualquer método de uma classe de métodos estatísticos, que se baseiam em amostragens aleatórias massivas para obter resultados numéricos), métodos matriciais para a solução de problemas de autovalores e autovetores, entre outros.

O objetivo da Física Computacional não é substituir teoria ou experimento, mas sim aumentar nossa compreensão acerca dos processos físicos. A aplicação de computadores para elevar o conhecimento sobre a natureza permite a ciência explorar fenômenos extremos, como as colisões de partículas subatômicas. Este conhecimento influencia diretamente o desenvolvimento de novas tecnologias que inspirarão os supercomputadores em um futuro muito próximo. Os modelos desenvolvidos a partir da Física Computacional vêm sendo empregados nos campos da biologia, química, engenharia, medicina, novos materiais, dispositivos eletrônicos, nanotecnologia e até mesmo no estudo de mercados financeiros. Princípios e métodos originais da Física são usados para gerar avanços em computação, com novos algoritmos e aplicações em computação quântica. O conhecimento em Ciência e Engenharia da Computação são utilizados como ferramenta para os avanços tanto em Física Experimental como Teórica, ao mesmo tempo em que conceitos da Física são aplicados à Teoria da Computação.

São desenvolvidos métodos para a mineração de dados, conhecida também pelo termo inglês *data mining* (Consiste em uma funcionalidade que agrega e organiza dados, encontrando neles padrões, associações, mudanças e anomalias relevantes), com o objetivo de reconhecer padrões e auxiliar a formulação de hipóteses a respeito de dados obtidos por meio da Física experimental. Para essa função, são muito utilizadas técnicas de Estatística, Inteligência artificial e Processamento de sinais. É estudada também a visualização computacional, com o objetivo de ajudar a interpretação dos dados, cada vez mais complexos, por parte do físico. Ela é utilizada tanto durante o processo de simulação, no trabalho de Física Teórica, como no processo de Mineração de dados (*data mining*).

Sendo assim, as simulações computacionais também podem ser usadas como ferramenta de exploração dos ilustres “experimentos pensados”, comumente usados em Física para checar ideias fundamentais e ilustrar suas consequências. Esta monografia está organizada na seguinte forma: No capítulo 2 apresentaremos o software Scilab, utilizado no desenvolvimento deste trabalho.

## 2 FUNDAMENTAÇÃO TEÓRICA

### 2.1 Software Scilab: Histórico e algumas ferramentas básicas de programação

O software Scilab começa sua história nos anos 80, com o Blaise, um software CACSD (Computer Aided Control System Design) criado no INRIA (Instituto Francês de Pesquisa em Ciência da Computação e Controle) e desenvolvido principalmente por François Delebecque e Serge Steer. Seu objetivo era fornecer uma ferramenta em Controle Automático para pesquisadores. Foi inspirado no software Matlab Fortran, desenvolvido por Cleve Moler, que mais tarde foi co-fundador da empresa John Little “The MathWorks”. Em 1984, o Blaise tornou-se Basile e foi distribuído durante alguns anos pela Simulog, a primeira startup do INRIA.

No início da década de 90, a Simulog parou de distribuir o Basile. O software passou a se chamar Scilab e foi então desenvolvido pelo INRIA, dentro do Grupo Scilab, composto pelos seis pesquisadores seguintes: Jean-Philippe Chancelier da ENPC (École Nationale des Ponts et Chaussées), François Delebecque, Claude Gomez, Maurice Goursat, Ramine Nikoukhah e Serge Steer do INRIA.

Então o INRIA decidiu distribuir o Scilab como software livre de código aberto. O Scilab 1.1, a primeira versão lançada do Scilab, foi colocado em um site de FTP anônimo em 2 de janeiro de 1994. O Scilab Group, com a colaboração ativa de desenvolvedores externos, desenvolveu o Scilab até o final de 2002 com a versão Scilab 2.7, distribuindo fontes e binários.

No início de 2003, para levar em consideração o aumento do número de pessoas que baixavam e utilizavam o Scilab, e para garantir seu futuro, desenvolvimento, manutenção, suporte e promoção, o INRIA decidiu criar o Consórcio Scilab com o apoio de empresas e organizações acadêmicas.

A empresa Scilab Enterprises é fundada em junho de 2010 com o apoio do INRIA, para garantir o futuro do Scilab. A empresa assumiu integralmente a edição e o desenvolvimento do Scilab desde julho de 2012, após 5 anos de versões do Scilab feitas pela Scilab Enterprises, a equipe operacional se junta ao Grupo ESI através da aquisição da empresa.

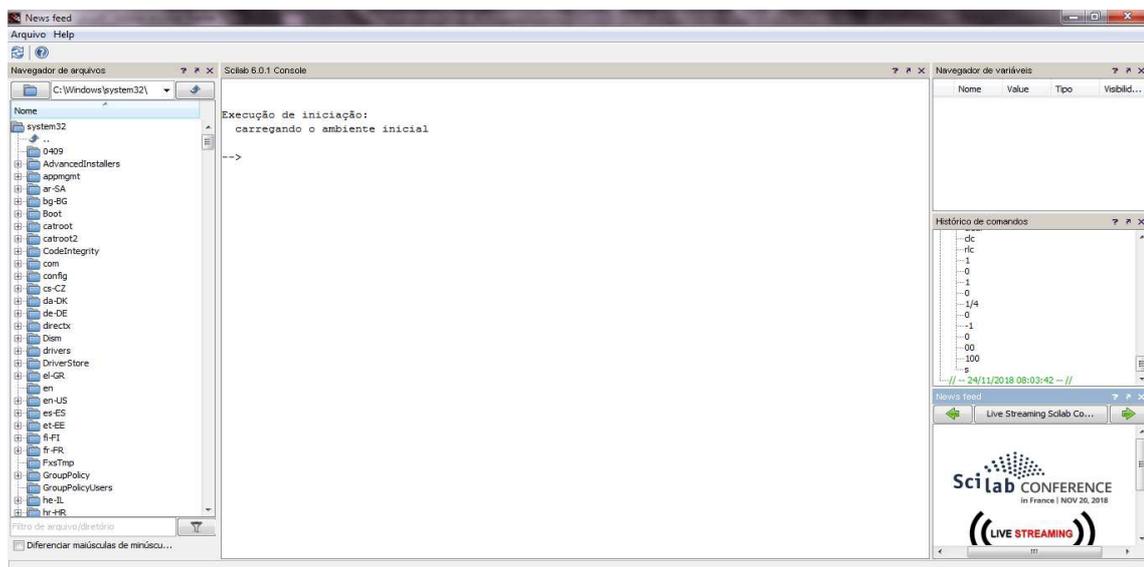
Scilab é uma linguagem de programação associada à uma rica coleção de algoritmos numéricos, que cobrem muitos aspectos de problemas de computação científica. Do ponto de vista de *software*, Scilab é uma linguagem interpretada. A linguagem Scilab permite a

compilação dinâmica e linkagem com outras linguagens como Fortran e C. Do ponto de vista de licença, Scilab é um software gratuito, disponível para Linux, Mac Os e Windows. Ele pode ser executado normalmente em muitas distribuições de Linux, e alternativamente, no site do Scilab é possível obter mais versões de binários e documentação para instalação. A ajuda online esta disponível em diversos idiomas, incluindo o português. Do ponto de vista científico, o Scilab começou focado em soluções computacionais para problemas de álgebra linear, mas, rapidamente, o número de aplicações se estendeu para muitas áreas da computação científica.

O Scilab é um software científico para computação numérica semelhante ao Matlab, que fornece um poderoso ambiente computacional aberto para aplicações científicas. Essa ferramenta é uma linguagem de programação de alto nível, orientada à análise numérica. A linguagem provê um ambiente para interpretação, com diversas ferramentas numéricas. Possui sofisticadas estruturas de dados, (incluindo listas, polinômios, sistemas lineares...), um interpretador de linguagem de programação de alto nível. O usuário pode definir novos tipos de informação e operações sobre esses tipos.

Dentre os softwares de análise científica, o Scilab pertence ao grupo de sistemas numéricos, cujo exemplo mais conhecido é o popular MATLAB que inspirou a criação de sistemas numéricos desenvolvidos em código livre como o GNU/Octave, FreeMat e o Scilab.

Figura 2: Interface do Scilab: Prompt de comando.



Fonte: Do próprio autor.

O Console do Scilab (Figura 2) é o ponto inicial para se entender o funcionamento do software. Nele encontramos o “prompt” →, que permite a execução de comandos, instruções,

carregamento de arquivos, configuração do path, iniciar outras janelas, entre outras possibilidades. No menu principal temos:

- Arquivo

Execute: Executa arquivos de scripts e com instruções de comando;

Abra um arquivo: Carrega arquivos de textos criados no ambiente do Scilab (.sce e.sci);

Mude o diretório corrente: Muda o diretório atual do ambiente para um novo local, comando: “->chdir(‘novaPasta’)” no prompt;

Mostre o diretório corrente: Mostra o diretório atual, “->pwd”; no prompt;

- Editar

No menu editar temos as opções padrão que dispensa comentários;

- Preferencias

Clear Console: Limpa o console mais não apaga os dados da memória, podemos executar esta função de outras duas formas: a) apertando a tecla F2 ou b) digitando “->clc”, no prompt;

- Controle

Resumo: Continua a execução após um pause ou um stop;

Aborte: Interrompe a execução;

Interruptor: Entra no modo de pause, podemos fazer “->Ctrl+C”; no prompt;

- Aplicações

SCINOTES: O editor de textos, podemos chamá-lo via prompt com o comando “->editor;

- Help

Abre o componente de ajuda.

- Scilab Demonstrações:

Exibe a janela de demonstrações de comandos do Scilab;

### 2.1.1 Comandos Básicos do Scilab

A forma de entrada é a tradicional, onde é necessário seguir algumas regras de sintaxe, ao contrário da forma intuitiva. As expressões matemáticas, operações e alguns objetos são referidos de maneiras tradicionais na computação, com parêntesis para separar expressões, barra simples para frações e divisões e acentos circunflexos para a representação de potências. Todas as formas tradicionais podem ser utilizadas também no modo intuitivo. Os comandos

são palavras-chave no Scilab e executam tarefas e instruções específicas, que são um conjunto de palavras-chave ordenadas digitadas no “prompt”. A finalização com o uso de “;” (ponto e vírgula) no final de cada instrução é opcional, sendo que, caso usemos a saída será omitida.

a) “clear”:

Este é um comando que cancela variáveis desprotegidas, removendo as variáveis nomeadas do ambiente. Assim, “clear” cancela todas as variáveis, exceto as que estão protegidas por “predef”. Os dois comandos, “predef(0)” e “clear”, removem todas as variáveis. Normalmente, variáveis protegidas são bibliotecas padrões e variáveis com o prefixo “%”.

b) “plot”:

Para gerar gráficos bidimensionais são necessários os comandos plot e plot2d. O comando plot realiza operações de plotagem de gráficos de forma mais simples e generalizada, já a função plot2d realiza uma plotagem de gráficos onde é possível variar o tipo de gráfico a ser utilizado.

c) “clc”:

Este comando limpa toda área de trabalho e coloca o cursor no início. Por padrão, esboços sucessivos são superpostos. Para limpar o esboço anterior, use “clf”.

d) “xtitle”:

“xtitle” adiciona títulos a um esboço 2d ou 3d. “title” é o título geral e “xlabel”, “ylabel” e “zlabel” são os títulos dos três eixos. Se os argumentos são matrizes, cada linha das matrizes é exibida em uma linha diferente.

e) “legend”:

Imprime uma legenda para o gráfico.

f) “xgrid([style])”:

O comando “xgrid” adiciona uma grade em um esboço 2d. O identificador style determina o tipo de tracejado ou a cor utilizada para o esboço da grade.

g) “for”:

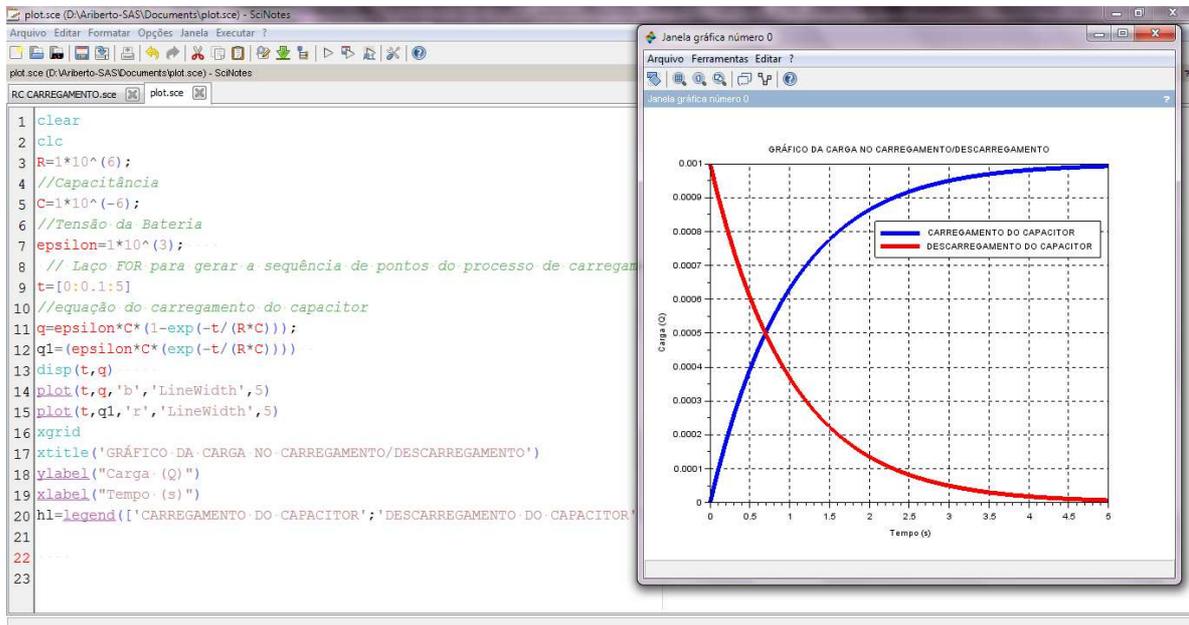
É a palavra-chave de linguagem de programação para laços (“loops”) (significa, literalmente “para...”). A sua sintaxe é:

for

variável = expressão, instrução ...

end

Figura 3: Exemplo do uso do comando “clear”, “clc”, “xgrid”, “xtitle”, “legend” e função “plot” (usando SCINOTES).



Fonte: Do próprio autor.

## 2.2 Método das diferenças finitas

No Método das Diferenças Finitas (MDF), o domínio contínuo do problema, é substituído por uma série de pontos discretos, nos quais são calculadas as incógnitas do problema.

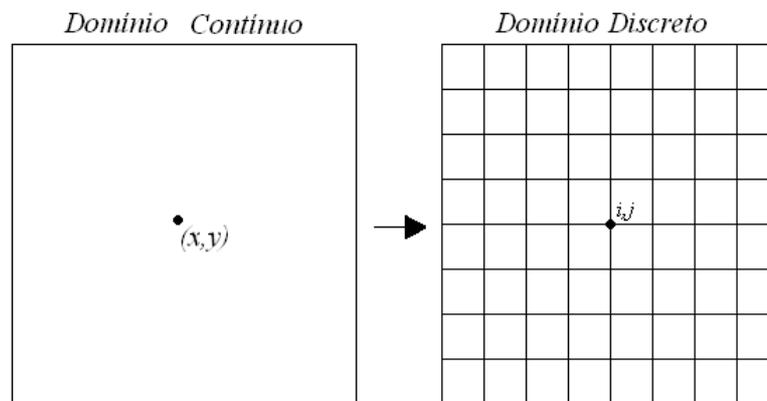
O método numérico das diferenças finitas é usado como uma abordagem alternativa para obter a aproximação da solução de uma equação diferencial parcial. A ideia básica desse método é transformar a resolução de uma equação diferencial em um sistema de equações algébricas, substituindo as derivadas por diferenças (RUGGIERO, 1996, p.357).

Nesse sentido, Franco (2006) enfatiza que,

O método numérico das diferenças finitas é facilmente executado em computadores. Ele consiste na discretização do domínio e na substituição das derivadas presentes na equação diferencial por aproximações utilizando apenas os valores numéricos da função. A ferramenta básica no cálculo das aproximações das derivadas é a fórmula de Taylor (p.432).

Essa substituição do contínuo pelo discreto denomina-se discretização, que é o processo de transferência de funções contínuas, modelos, variáveis e equações em contrapartes discretas. Podemos visualizar o processo de discretização do espaço computacional, reservado a modelagem de um processo qualquer, na figura 4.

Figura 4: Exemplo de mudança do domínio contínuo de coordenadas  $(x,y)$  para o discreto de coordenadas  $(i,j)$ .



Fonte: Do próprio autor.

Uma vez efetuada a discretização do domínio do problema, aplica-se o método de diferenças finitas para a determinação das incógnitas. As derivadas, que aparecem na equação original, são substituídas por fórmulas discretas de diferenças. A aplicação dessas fórmulas aos pontos do domínio discretizado gera um sistema de equações algébricas, cuja solução fornece os valores das incógnitas do problema nesses pontos discretos.

### 2.2.1 Derivadas de 1ª ordem

Antes de apresentar as aplicações do processo de discretização do método das diferenças finitas é necessário desenvolver todas as aproximações utilizadas nos modelos estudados no desenvolvimento desse trabalho.

Por definição a derivada de uma função  $u(x)$  em um ponto  $x_n$  é dada por:

$$\left. \frac{du}{dx} \right)_{x=x_n} = \lim_{h \rightarrow 0} \frac{u(x_n + h) - u(x_n)}{h}, \quad (1.1)$$

onde  $h$  é um incremento. Para pequenos valores de  $h$  a equação (1.1) poder ser aproximada, porém finito, pela seguinte estrutura.

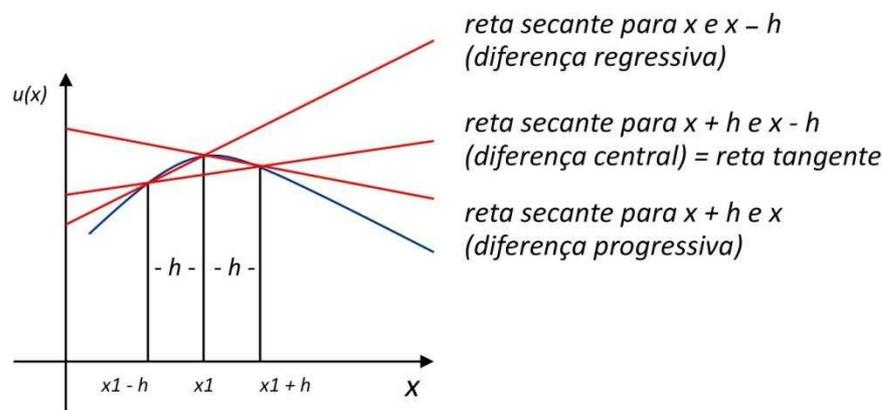
$$\left. \frac{du}{dx} \right)_{x=x_n} \cong \frac{u(x_n + h) - u(x_n)}{h}, \quad (1.2)$$

A aproximação definida pela equação (1.2) é denominada diferença progressiva porque utiliza um ponto à frente de  $x_n$ , ou seja, ponto  $(x_n + h)$ . De maneira análoga, pode-se definir uma diferença regressiva.

$$\left. \frac{du}{dx} \right)_{x=x_n} \cong \frac{u(x_n) - u(x_n - h)}{h}. \quad (1.3)$$

A figura 5 mostra a influência das diferentes derivadas tomadas em relação ao ponto  $x$  e sua vizinhança à direita e a esquerda.

Figura 5: Visualização dos limites laterais para uma determinada função contínua  $u(x)$ .



Fonte: Do próprio autor.

A figura 5 ainda apresenta outra forma de aproximação, que é a diferença central:

$$\left. \frac{du}{dx} \right)_{x=x_n} \cong \frac{u(x_n + h) - u(x_n - h)}{2h}. \quad (1.4)$$

As fórmulas aproximadas, (1.2), (1.3), (1.4) também podem ser obtidas, através de *Séries de Taylor* truncadas, o que permite contabilizar o erro cometido nas aproximações.

### 2.2.1.1 Diferença progressiva

A expansão em *Série de Taylor* do valor de  $u$  em  $x_n = x_n + h$  em torno do valor de  $u$  em  $x = x_n$  é:

$$u(x_n + h) \cong u(x_n) + h \left. \frac{du}{dx} \right)_{x=x_n} + \frac{h^2}{2!} \left. \frac{d^2u}{dx^2} \right)_{x=x_n} + \frac{h^3}{3!} \left. \frac{d^3u}{dx^3} \right)_{x=x_n} + \dots \quad (1.5)$$

A expressão (1.5), encontramos a forma:

$$\left. \frac{du}{dx} \right)_{x=x_n} \cong \frac{u(x_n + h) - u(x_n)}{h} - \frac{h}{2!} \left. \frac{d^2u}{dx^2} \right)_{x=x_n} + \frac{h^2}{6} \left. \frac{d^3u}{dx^3} \right)_{x=x_n} \dots \quad (1.6)$$

Uma vez que admitimos  $h$  é pequeno, podemos omitir a série no ponto escolhido. Desprezando os termos relativos às derivadas de ordem igual ou superior a dois. Obtemos assim, a expressão da diferença progressiva. O maior termo desprezado é igual ao produto de  $h$  por uma constante, ou seja, é da ordem de  $h$  (notação  $O(h)$ ). O erro que se comete quando se substitui a expressão (1.1) pela expressão (1.2) é  $O(h)$ .

### 2.2.1.2 Diferença regressiva

Analogamente, a expressão em série de Taylor do valor de  $u$  em  $x = x_n - h$ , em torno do valor de  $u$  em  $x = x_n$ , é:

$$u(x_n - h) \cong u(x_n) - h \left. \frac{du}{dx} \right)_{x=x_n} + \frac{h^2}{2!} \left. \frac{d^2u}{dx^2} \right)_{x=x_n} - \frac{h^3}{3!} \left. \frac{d^3u}{dx^3} \right)_{x=x_n} + \dots \quad (1.7)$$

Resolvendo-se, a equação (1.7), para  $\left. \frac{du}{dx} \right)_{x=x_n}$ , obtemos:

$$\left. \frac{du}{dx} \right)_{x=x_n} \cong \frac{u(x_n) - u(x_n - h)}{h} + \frac{h}{2!} \left. \frac{d^2u}{dx^2} \right)_{x=x_n} - \frac{h^2}{6} \left. \frac{d^3u}{dx^3} \right)_{x=x_n} \dots \quad (1.8)$$

Ao se desprezar os termos relativos às derivadas de ordem dois ou superiores, obtém-se a expressão da diferença regressiva, que também introduz um erro  $O(h)$ , quando se substitui (1.1).

### 2.2.1.3 Diferença central

Subtraindo-se (1.5) de (1.7) tem-se:

$$u(x_n + h) - u(x_n - h) = 2h \left. \frac{du}{dx} \right)_{x=x_n} + \frac{2h^3}{3!} \left. \frac{d^3u}{dx^3} \right)_{x=x_n} + \dots, \quad (1.9)$$

e resolvendo a expressão (1.9), em termos de  $\left. \frac{du}{dx} \right)_{x=x_n}$ , obtemos:

$$\left. \frac{du}{dx} \right)_{x=x_n} \cong \frac{u(x_n + h) - u(x_n - h)}{2h} - \frac{h^2}{6} \left. \frac{d^3u}{dx^3} \right)_{x=x_n}. \quad (1.10)$$

Para se obter a expressão correspondente à diferença central são desprezados os termos relativos às derivadas de ordem 3 e superiores, conseqüentemente o *erro* cometido quando se substitui (1.1) por (1.4) é da ordem de  $O(h^2)$ . Este erro é menor do que aquele cometido pelo uso da diferença regressiva e progressiva conforme mostra a Figura 5.

### 2.2.2 Derivadas de 2ª ordem

As derivadas de segunda ordem também podem ser adquiridas através de *Séries de Taylor*. As expressões (1.5) e (1.7) podem ser somadas, resultando em:

$$u(x_n + h) - u(x_n - h) = 2u(x_n) + h \left. \frac{d^2u}{dx^2} \right)_{x=x_n} + \frac{2h^4}{4!} \left. \frac{d^4u}{dx^4} \right)_{x=x_n} + \dots \quad (1.11)$$

Resolvendo para  $\left. \frac{d^2u}{dx^2} \right)_{x=x_n}$  temos:

$$\left. \frac{d^2u}{dx^2} \right)_{x=x_n} \cong \frac{u(x_n + h) - 2u(x_n) + u(x_n - h)}{h^2} - \frac{h^2}{12} \left. \frac{d^4u}{dx^4} \right)_{x=x_n} + \dots \quad (1.12)$$

A aproximação para a derivada segunda é uma aproximação do tipo diferença central e o erro cometido é da ordem de  $O(h^2)$ :

$$\left. \frac{d^2u}{dx^2} \right)_{x=x_n} \cong \frac{u(x_n + h) - 2u(x_n) + u(x_n - h)}{h^2}. \quad (1.13)$$

### 3 APLICAÇÃO: CIRCUITOS ELÉTRICOS RC E RLC

Neste capítulo trataremos das deduções das equações de carga e descarga de um capacitor, bem como da relação da tensão e corrente do circuito RLC. Utilizaremos para isto o software Scilab (versão 6.0.1), com a modelagem numérica do Método de Diferenças Finitas.

#### 3.1 Dedução das equações de carga e descarga do capacitor em um circuito RC

Para o equacionamento da equação correspondente à carga do capacitor, suponha que com o capacitor totalmente descarregado, a chave mostrada no circuito abaixo é colocada na posição um conectando a fonte  $\varepsilon$  ao circuito.

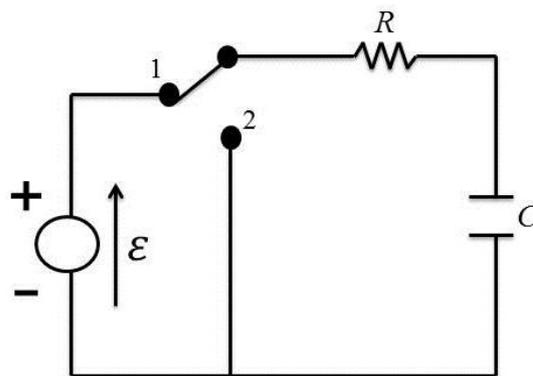


Figura 6: Circuito RC ligado a uma bateria.

Fonte: Do próprio autor.

Nessas condições, sabendo que a queda de tensão no resistor é dada pela lei de Ohm:

$$V_R = R i(t), \quad (2.1)$$

e que a tensão no capacitor é por definição:

$$V_C = \frac{q(t)}{C}, \quad (2.2)$$

e que a corrente elétrica corresponde à variação de carga no tempo:

$$i = \frac{dq(t)}{dt}, \quad (2.3)$$

pela lei das malhas de Kirchhoff, podemos equacionar o problema do carregamento do capacitor da seguinte maneira:

$$V_C + V_R = \varepsilon, \quad (2.4)$$

ou ainda,

$$Ri + \frac{q(t)}{C} = \varepsilon. \quad (2.5)$$

Substituindo a equação (2.3) em (2.5), temos:

$$R \frac{dq(t)}{dt} + \frac{q(t)}{C} = \varepsilon. \quad (2.6)$$

Para resolvermos essa equação diferencial de primeira ordem utilizaremos o método de separação de variáveis e, assim, isolaremos  $q(t)$ , dividimos agora a equação (2.6) por  $R$ .

$$\frac{dq(t)}{dt} + \frac{q(t)}{RC} = \frac{\varepsilon}{R}. \quad (2.7)$$

Como o termo  $RC$  é constante de tempo  $\tau$  do circuito, temos,

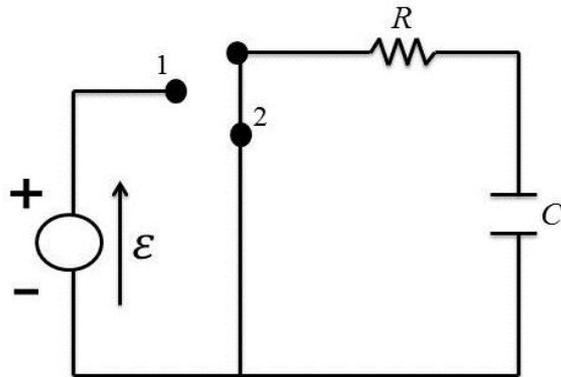
$$\frac{dq(t)}{dt} + \frac{q(t)}{\tau} = \frac{\varepsilon}{R}. \quad (2.8)$$

que tem como solução geral,

$$q(t) = \varepsilon C \left( 1 - e^{-\frac{t}{\tau}} \right). \quad (2.9)$$

Para o equacionamento da fórmula correspondente a descarga do capacitor, iremos assumir agora que o capacitor já está totalmente carregado e, assim, passaremos a chave para a posição dois, onde a fonte é desconectada do circuito e o capacitor será descarregado sobre o resistor.

Figura 7: Circuito RC com a retirada da bateria.



Fonte: Do próprio autor.

Nessas condições, analisando a nova malha pela lei das malhas de kirchhoff, e trabalhando matematicamente a expressão, como fizemos anteriormente no caso da carga do capacitor, chegaremos a seguinte EDO:

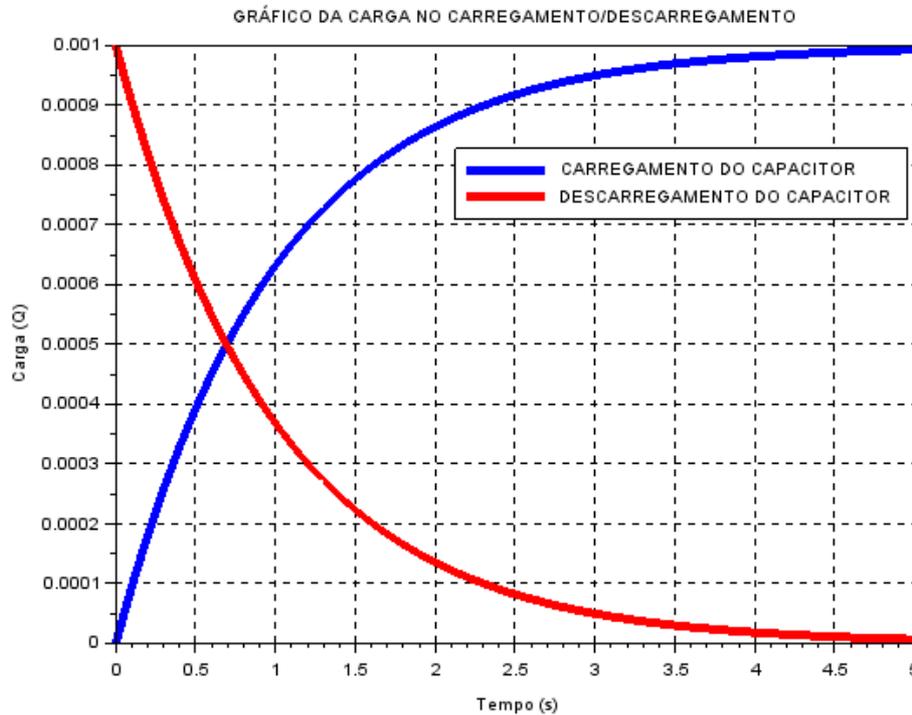
$$\frac{dq(t)}{dt} + \frac{q(t)}{\tau} = 0, \quad (2.10)$$

que tem como solução geral,

$$q(t) = \varepsilon C \left( e^{-\frac{t}{\tau}} \right). \quad (2.11)$$

A figura 8 mostra o processo de carregamento e descarregamento do capacitor de acordo com as soluções gerais, equações (2.9) e (2.11).

Figura 8: Gráfico do carregamento/descarregamento do capacitor.



Com os seguintes valores,  $R=1.10^6\Omega$ ,  $C=1.10^6F$  e  $\varepsilon=1.10^3V$ . Fonte: Do próprio autor.

Assim, objetivamos encontrar um resultado gráfico semelhante no processo de discretização que valide a análise numérica e o programa desenvolvido nesse trabalho.

### 3.1.1 Circuito RC - Modelagem numérica com diferenças finitas

#### a) Processo de carregamento de um capacitor.

Utilizando o operador de diferenças finitas de 1ª ordem, equação (2.7), para o processo de carregamento, temos,

$$\frac{q(t+1) - q(t)}{h} + \frac{q(t)}{RC} = \frac{\varepsilon}{R} \quad (2.12)$$

Resolvendo a equação (2.12) para  $q(t+1)$  temos o esquema geral,

$$q(t+1) = h \frac{\varepsilon}{R} - \frac{q(t)h}{RC} + q(t), \quad (2.13)$$

ou simplesmente,

$$q(t + 1) = h \frac{\varepsilon}{R} - q(t)h \left( \frac{1}{RC} - \frac{1}{h} \right). \quad (2.14)$$

A equação (2.14) gera uma sequência de pontos, que ao serem plotados deverão apresentar um comportamento compatível com a solução geral do carregamento do capacitor.

b) Processo de descarregamento de um capacitor.

Iremos obter a equação do descarregamento do capacitor, em um processo análogo usado para desenvolver a equação (2.15) substituindo a equação (1.2) na equação (2.10), temos,

$$\frac{q(t + 1) - q(t)}{h} + \frac{q(t)}{RC} = 0. \quad (2.15)$$

Resolvendo para  $q(t + 1)$  temos o esquema geral,

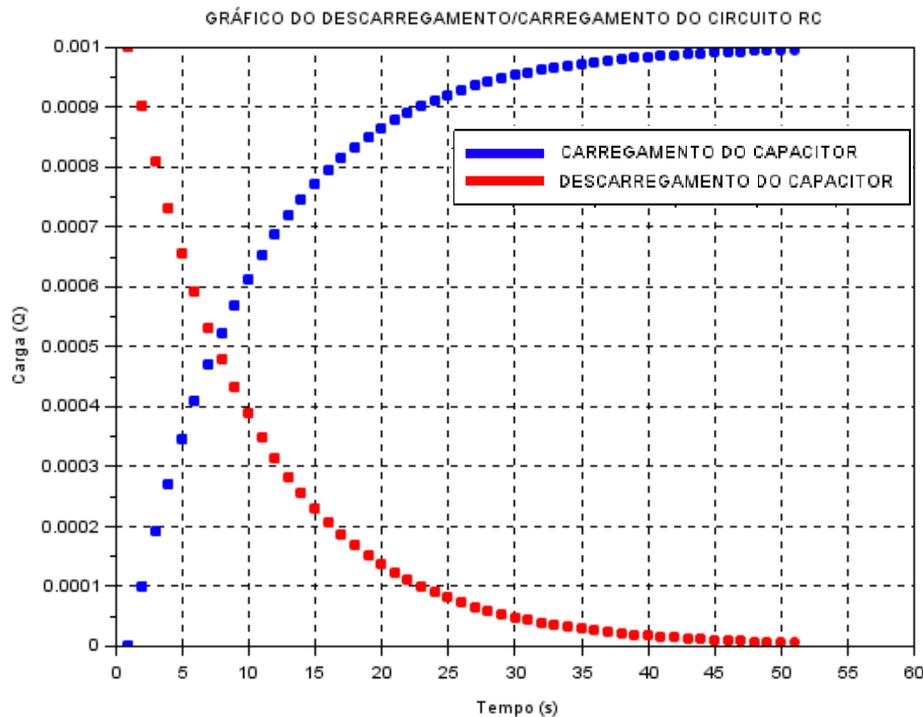
$$q(t + 1) = -\frac{q(t)h}{RC} + \frac{q(t)h}{h}, \quad (2.16)$$

ou simplesmente,

$$q(t + 1) = -q(t)h + \left( \frac{1}{RC} - \frac{1}{h} \right), \quad (2.17)$$

Obtemos assim, a equação discreta equivalente à solução geral, equação (2.11), do problema do descarregamento do capacitor.

Figura 9: Gráfico do carregamento/descarregamento do capacitor, utilizando o método de diferenças finitas.



Com os seguintes valores,  $R=1.10^6\Omega$ ,  $C=1.10^6F$  e  $\mathcal{E}=1.10^3V$ . Fonte: Do próprio autor.

A figura 9 mostra a equivalência com o modelo teórico, um comportamento tipicamente exponencial nos processos de carregamento e descarregamento do capacitor. O gráfico foi obtido pelo programa desenvolvido para o estudo do MDF, em anexo.

### 3.2 Circuito RLC: Relação entre tensão e corrente

Neste capítulo iremos discutir as formas de encontrar as respostas naturais dos circuitos RLC, em série e paralelo, respectivamente. As equações que modelam os circuitos RLC em série e paralelo para os casos de resposta natural serão deduzidas por intermédio da aplicação das leis de kirchoff das correntes e das tensões. Para isso, se faz necessário ter o conhecimento prévio sobre a relação entre corrente e tensão nos elementos de circuito implicados.

Sabemos que o capacitor é um elemento de circuito que se opõe a variação de tensão, a relação entre tensão e corrente nesse elemento é,

$$i_c(t) = C \frac{dV_C(t)}{dt}, \quad (3.1)$$

onde  $i_C(t)$  é a corrente no tempo e  $V_C(t)$  a tensão. Essas duas funções estão relacionadas por meio de uma constante denominada capacitância,  $C$ .

O indutor é uma parte do circuito que se opõe a variação de corrente. A relação entre tensão  $V_L(t)$  e corrente  $i_L(t)$  para esse elemento é derivada da Lei de Faraday, e dada por,

$$V_L(t) = L \frac{di_C(t)}{dt}, \quad (3.2)$$

sendo  $L$  a constante que relaciona variação de corrente com tensão, denominada indutância. O resistor é um elemento que dissipa energia mediante o efeito Joule, pela lei de Ohm, tensão e corrente no resistor é relacionado por,

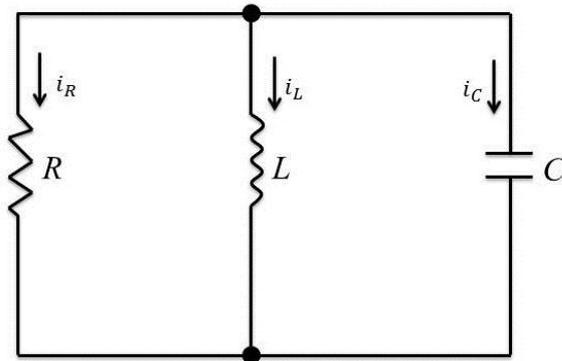
$$V_R(t) = Ri(t), \quad (3.3)$$

onde a resistência  $R$  é uma constante que relaciona tensão e corrente, proporcionalmente. Fisicamente o capacitor é um conjunto de dois ou mais condutores isolados entre si por meio de dielétricos e que tem como função armazenar carga e energia elétrica no campo eletrostático que se estabelece entre os condutores. Analogamente o indutor armazena energia magnética no seu enrolamento. A partir dessas relações já podemos iniciar o estudo do comportamento transitório do circuito RLC.

### 3.2.1 Circuito RLC paralelo: Resposta natural

A configuração do circuito RLC paralelo é mostrada na Figura 10. Circuitos dessa natureza tem a mesma tensão nos terminais de todos os dispositivos, por isso, a resposta natural será dada em função da tensão no capacitor. Considerando que alguma energia já havia sido armazenada nesse circuito (no indutor ou capacitor), então, em  $t = 0$  temos a corrente no indutor  $i_L(0) = I_0$  e tensão no capacitor  $V_C(0) = V_0$ .

Figura 10: Circuito RLC paralelo (sem fonte).



Fonte: Do próprio autor.

Utilizando a lei de Kirchhoff das correntes e realizando algumas manipulações algébricas tem-se,

$$\frac{d^2}{dt^2}V(t) + \frac{1}{RC} \frac{d}{dt}V(t) + \frac{1}{LC}V(t) = 0. \quad (3.4)$$

Onde consideremos uma solução do tipo;

$$V(t) = Ae^{\lambda_1 t}. \quad (3.5)$$

Substituindo a equação (3.5) em (3.4) e realizando as derivações, obtemos a equação característica;

$$\lambda^2 + \frac{1}{RL}\lambda + \frac{1}{LC} = 0, \quad (3.6)$$

cujas raízes da equação (3.6) são,

$$\lambda = -\frac{1}{2RC} \pm \sqrt{\left(\frac{1}{2RC}\right)^2 - \frac{1}{LC}}. \quad (3.7)$$

Definimos  $\alpha = 1/2RC$  e  $\omega_0 = 1/\sqrt{LC}$ , logo, podemos reescrever a equação (3.7) como,

$$\lambda_{1,2} = -\alpha \pm \sqrt{\alpha^2 - \omega_0^2}. \quad (3.8)$$

Desta forma, além das duas soluções possíveis,  $V(t) = A_1 e^{\lambda_1 t}$  e  $V(t) = A_2 e^{\lambda_2 t}$ , mas também a soma delas também é válida. Veja que  $\lambda_1$  e  $\lambda_2$  dependem dos parâmetros do circuito (resistência  $R$ , indutância  $L$  e capacitância  $C$ ). Na solução proposta, os valores das raízes da equação característica estão no argumento da função exponencial, que deve ser adimensional. Como surgem multiplicadas pelo tempo, sua, dimensões devem ser o inverso do tempo, ou seja, dimensão de frequência.

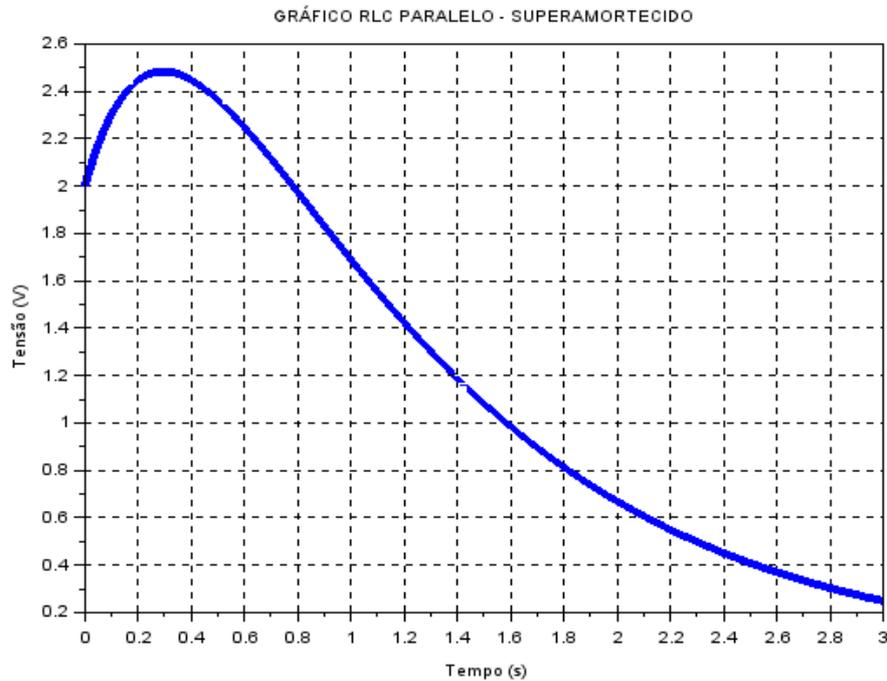
Uma vez que  $\lambda_1$  e  $\lambda_2$  são frequências,  $\alpha$  e  $\omega_0$  também são. Para distingui-las, denominamos  $\lambda_1$  e  $\lambda_2$  como frequências complexas,  $\alpha$  frequência de Neper e  $\omega_0$  frequência angular de ressonância. Veja ainda que dependendo dos valores dos parâmetros do circuito as raízes da equação característica podem ser reais, imaginárias ou complexas. Há três casos que precisam ser considerados:

1.  $\alpha > \omega_0$  - Raízes Reais e Distintas: Caso Superamortecido.

A solução geral da equação diferencial de 2ª ordem (3.1) tem a forma,

$$V(t) = A_1 e^{\lambda_1 t} + A_2 e^{\lambda_2 t}. \quad (3.9)$$

Figura 11: Gráfico do circuito RLC Paralelo (Caso Superamortecido).



Com os seguintes valores,  $R=1\Omega$ ,  $L=4/3H$ ,  $C=1/4F$ ,  $I_0=3A$  e  $V_0=2V$ . Encontramos os valores das constantes,  $A_1 = 5$  e  $A_2 = -3$ , numericamente. Fonte: Do próprio autor.

## 2. $\alpha < \omega_0$ - Raízes Complexas: Caso Subamortecido.

Por conveniência, expressamos as raízes como,

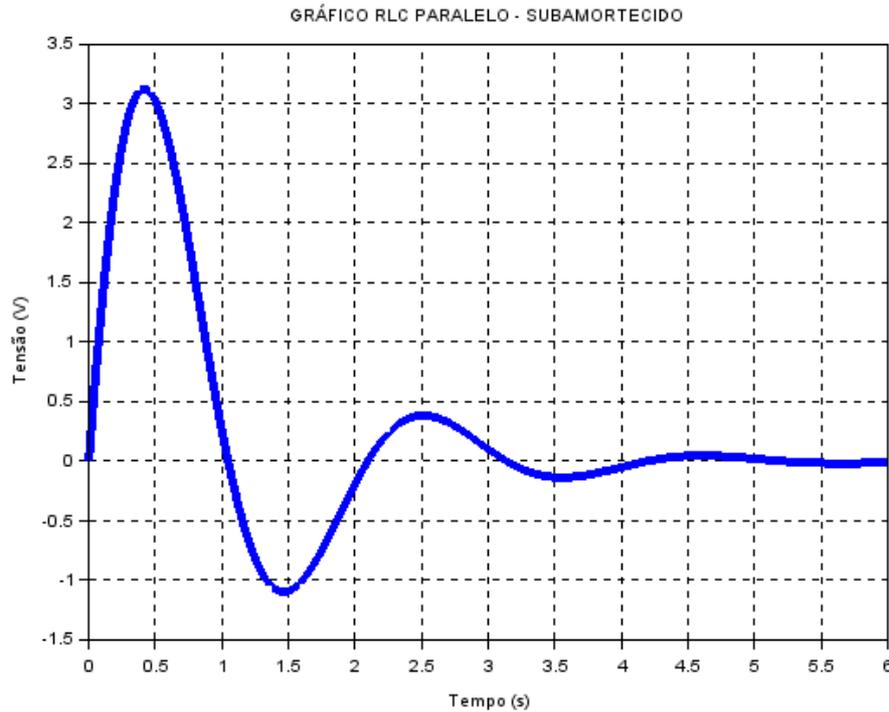
$$\lambda_{1,2} = -\alpha \pm j\omega_d, \quad (3.10)$$

onde  $j = \sqrt{-1}$  e  $\omega_d = \sqrt{\omega_0^2 - \alpha^2}$  a frequência de oscilação. Substituindo esses valores na solução geral (equação (3.9)), obtemos,

$$V(t) = e^{-\alpha t} [D \cos(\omega_d t) + K \text{sen}(\omega_d t)]. \quad (3.11)$$

A tensão oscila diminuindo sua amplitude de acordo com o fator exponencial, dessa forma,  $\alpha$  também é chamado de fator de amortecimento.

Figura 12: Gráfico do circuito RLC Paralelo (Caso Subamortecido).



Com os seguintes valores,  $R=5\Omega$ ,  $L=1\text{H}$ ,  $C=1/10\text{F}$ ,  $I_0=3/2\text{A}$  e  $V_0=0\text{V}$ . Encontramos os valores das constantes,  $D = 0$  e  $K = 5$ , numericamente. Fonte: Do próprio autor.

### 3. $\alpha = \omega_0$ - Raízes Reais e Iguais: Caso de Amortecimento Crítico.

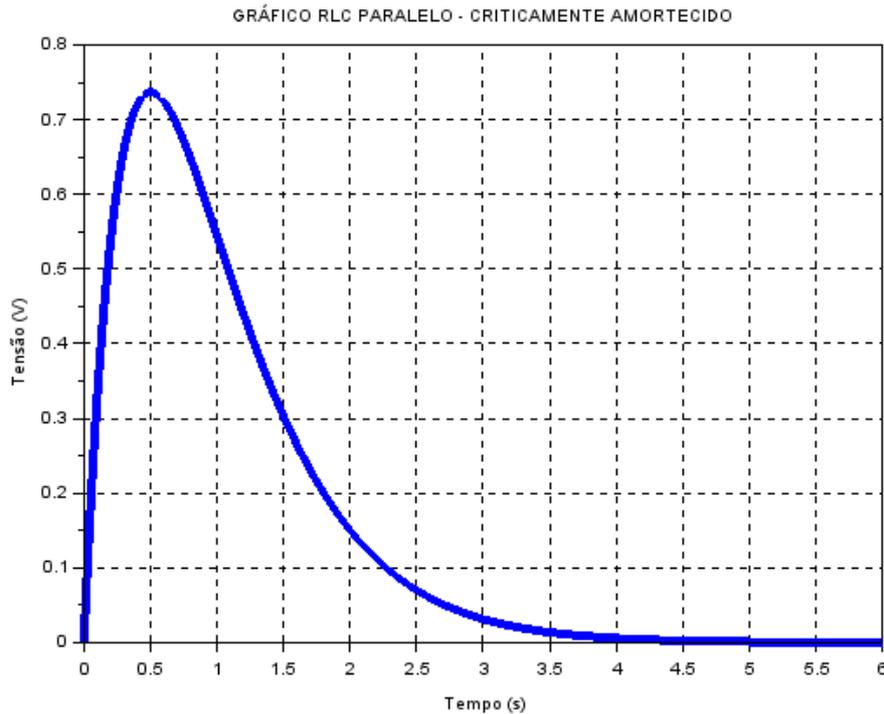
As frequências naturais são reais e iguais, dadas por,  $\lambda_1 = \lambda_2 = -\alpha$ , assim,  $V_C = (B_1 + B_2) e^{-\alpha t}$ . Essa solução não satisfaz duas condições iniciais, um caminho é agregar uma constante e fazer um produto com o tempo, que também é solução da EDO,

$$V(t) = (B_1 + B_2 t) e^{-\alpha t} \quad (3.12)$$

Admitindo as condições iniciais,

$$\frac{dV_C(0)}{dt} = \frac{i_C(0)}{C}, \quad \frac{di_L(0)}{dt} = \frac{V_L(0)}{L}. \quad (3.13)$$

Figura 13: Gráfico do circuito RLC Paralelo (Caso Criticamente amortecido).

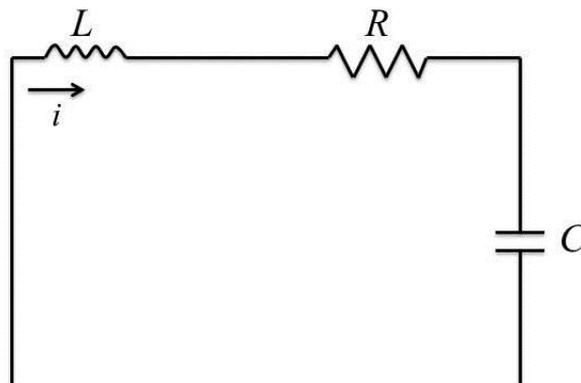


Com os seguintes valores,  $R=1\Omega$ ,  $L=1\text{H}$ ,  $C=1/4\text{F}$ ,  $I_0=-1\text{A}$  e  $V_0=0\text{V}$ . Encontramos os valores das constantes,  $B_1 = 0$  e  $B_2 = 4$ , numericamente. Fonte: Do próprio autor.

### 3.2.2 Circuito RLC série: Resposta natural

O circuito RLC série é mostrado na Figura 14. De modo semelhante ao realizado para o RLC paralelo, utiliza-se a lei de Kirchhoff.

Figura 14: Circuito RLC série (sem fonte).



Fonte: Do próprio autor.

Aplicando a lei de Kirchhoff das tensões e realizando algumas manipulações algébricas, tem-se a equação diferencial para RLC série:

$$\frac{d^2}{dt^2}I(t) + \frac{R}{L}\frac{d}{dt}I(t) + \frac{1}{LC}I(t) = 0. \quad (3.14)$$

Veja que a estrutura da equação diferencial acima é equivalente à equação diferencial do circuito RLC paralelo. Substituindo a solução geral na equação diferencial de 2ª ordem e efetuando as derivações, encontramos a equação característica,

$$\lambda^2 + \frac{R}{L}\lambda + \frac{1}{LC} = 0, \quad (3.15)$$

cujas, suas raízes são,

$$\lambda = -\frac{R}{2L} \pm \sqrt{\left(\frac{R}{2L}\right)^2 - \frac{1}{LC}}, \quad (3.16)$$

onde definimos  $\alpha = R/2L$  e  $\omega_0 = 1/\sqrt{LC}$ , logo,

$$\lambda_{1,2} = -\alpha \pm \sqrt{\alpha^2 - \omega_0^2}. \quad (3.17)$$

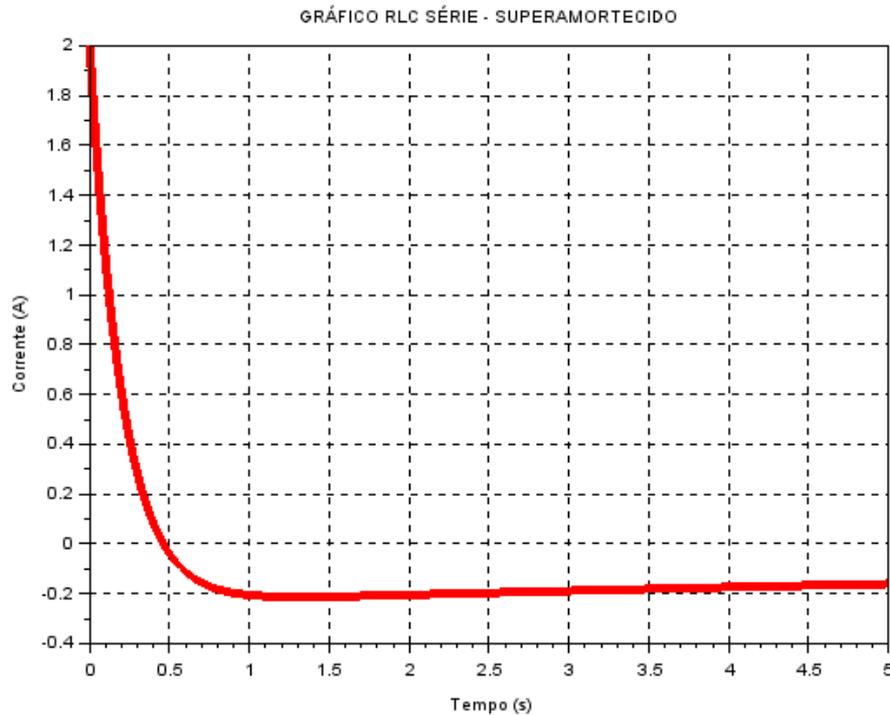
Desta forma, a única diferença em relação ao circuito RLC paralelo é o valor da frequência de Neper,  $R/2L$ . Como nos três casos discutidos no circuito RLC paralelo, a equação diferencial que descreve o circuito RLC série admite as seguintes soluções:

1.  $\alpha > \omega_0$ - Raízes Reais e Distintas: Caso Superamortecido.

A solução geral da equação (3.1) tem a forma:

$$I(t) = A_1e^{\lambda_1 t} + A_2e^{\lambda_2 t}. \quad (3.18)$$

Figura 15: Gráfico do circuito RLC Série (Caso Superamortecido).



Com os seguintes valores,  $R=50\Omega$ ,  $L=10\text{H}$ ,  $C=1/4\text{F}$ ,  $I_0=2\text{A}$  e  $V_0=10\text{V}$ . Encontramos os valores das constantes,  $A_1 = -0,24034734$  e  $A_2 = 2,24034734$ . Fonte: Do próprio autor.

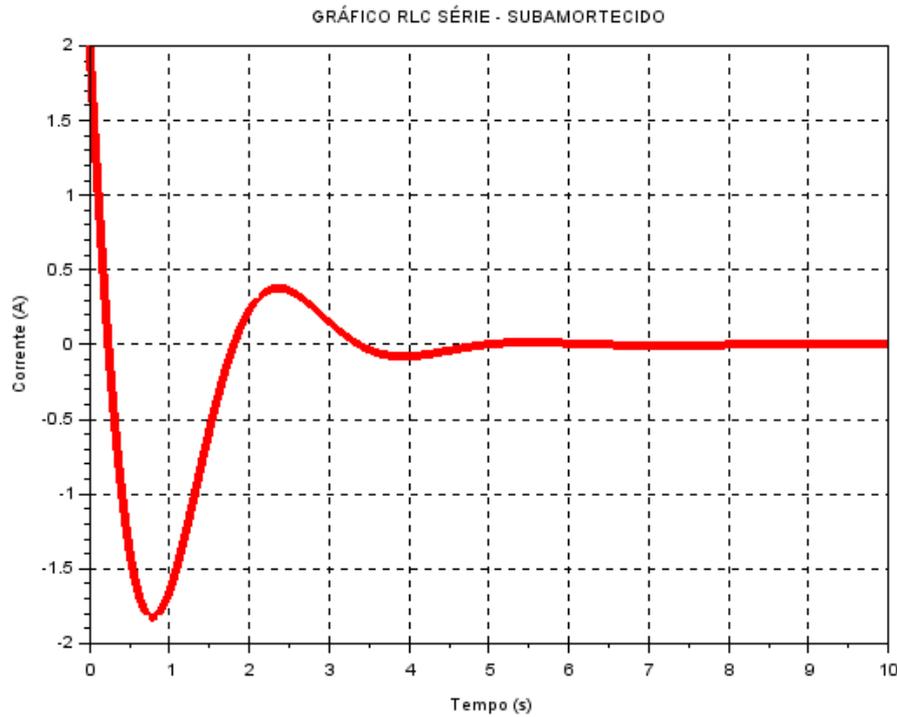
2.  $\alpha < \omega_0$ - Raízes Complexas: Caso Subamortecido.

$$\lambda_{1,2} = -\alpha \pm j\omega_d, \quad (3.19)$$

onde  $j = \sqrt{-1}$  e  $\omega_d = \sqrt{\omega_0^2 - \alpha^2}$  a frequência de oscilação. Substituindo esses valores na solução geral e realizando algumas transformações trigonométricas obtemos,

$$I(t) = e^{-\alpha t} [D \cos(\omega_d t) + K \text{sen}(\omega_d t)]. \quad (3.20)$$

Figura 16: Gráfico do circuito RLC Série (Caso Subamortecido).



Com os seguintes valores,  $R=2\Omega$ ,  $L=1\text{H}$ ,  $C=1/5\text{F}$ ,  $I_0=2\text{A}$  e  $V_0=6\text{V}$ . Encontramos os valores das constantes,  $D = -4$  e  $K = 2$ , numericamente. Fonte: Do próprio autor.

3.  $\alpha = \omega_0$  - Raízes Reais e Iguais: Caso de Amortecimento Crítico.

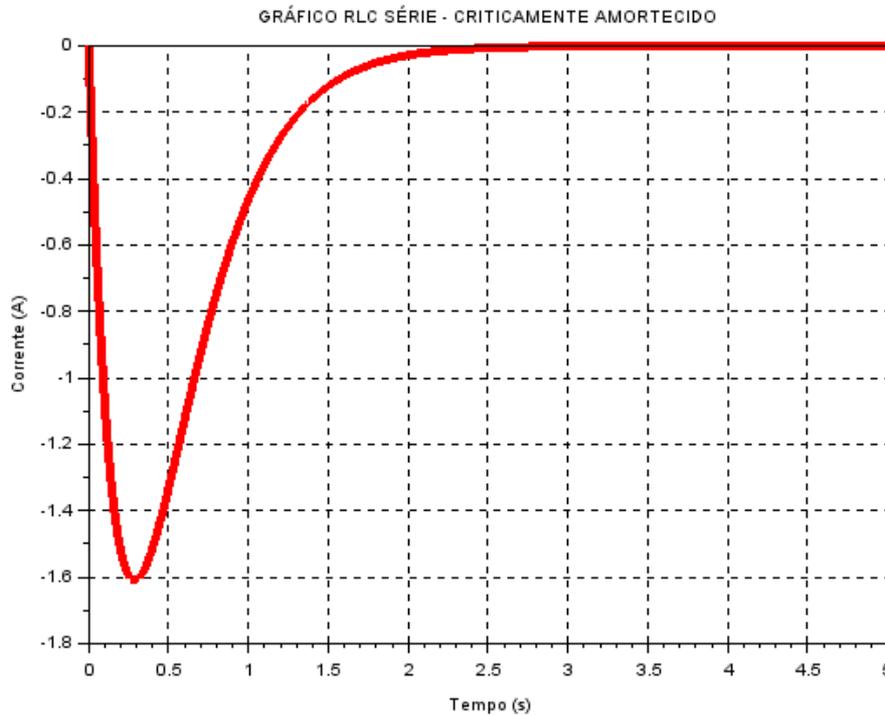
A solução geral assume a seguinte forma

$$I(t) = (B_1 + B_2 t)e^{-\alpha t}. \quad (3.21)$$

Com as condições iniciais,

$$\frac{dV_C(0)}{dt} = \frac{i_C(0)}{C}, \quad \frac{di_L(0)}{dt} = \frac{V_L(0)}{L}. \quad (3.22)$$

Figura 17: Gráfico do circuito RLC Série (Caso Criticamente amortecido).



Com os seguintes valores,  $R=7\Omega$ ,  $L=1H$ ,  $C=4/49F$ ,  $I_0=0A$  e  $V_0=10V$ . Encontramos os valores das constantes,  $B_1 = 0$  e  $B_2 = -10$ , numericamente. Fonte: Do próprio autor.

Na próxima seção, aplicaremos o método das diferenças finitas para discretizar as equações diferenciais de 2ª ordem do circuito RLC.

### 3.3 Circuito RLC paralelo e série: Modelagem numérica

#### 3.3.1 Circuito RLC paralelo

Inicialmente definimos os operadores de 1ª e 2ª ordem, em termos da tensão, de acordo com o método das diferenças finitas.

$$\frac{d^2V}{dt^2} = \frac{V(n+1) - 2V(n) + V(n-1)}{\Delta t^2}, \quad (3.23)$$

$$\frac{dV}{dt} = \frac{V(n+1) - V(n)}{\Delta t}. \quad (3.24)$$

Fazendo  $\Delta t = h$ , podemos escrever a equação diferencial do circuito RLC paralelo em diferenças finitas como,

$$\frac{V(n+1) - 2V(n) + V(n-1)}{h^2} + \frac{1}{RC} \frac{V(n+1) - V(n)}{h} + \frac{1}{LC} V(n) = 0. \quad (3.25)$$

Agora, resolvendo a equação (3.25) em termos de  $V(n+1)$ ,

$$V(n+1) = \frac{2V(n) - V(n-1)}{Ah^2} + V(n) \frac{B}{A}, \quad (3.26)$$

onde utilizamos a seguinte simplificação,

$$A = \left( \frac{1}{h^2} + \frac{1}{RCh} \right), B = \left( \frac{1}{RCh} - \frac{1}{LC} \right). \quad (3.27)$$

No ponto de vista computacional, temos um problema de contorno envolvendo  $V(n-1)$ , que pode ser resolvido utilizando a condição inicial que nos permite encontrar a resposta natural do circuito RLC,

$$\frac{dV(0)}{dt} = \frac{I_0}{C}, \quad (3.28)$$

ou em termos de diferenças finitas,

$$\frac{V(n) - V(n-1)}{h} = \frac{I_0}{C}, \quad (3.29)$$

logo,

$$V(n-1) = V(n) - \frac{I_0}{C}. \quad (3.30)$$

Substituindo a equação (3.30) na equação (3.26), temos que,

$$V(n+1) = \frac{1}{Ah^2} \left( V(n) + h \frac{I_0}{C} \right) + V(n) \frac{B}{A}, \quad (3.31)$$

que nos garante a condição inicial que assegura o êxito e estabilidade da simulação computacional, com  $n=0$

$$V(1) = \frac{1}{Ah^2} \left( V_0 + h \frac{I_0}{C} \right) + V_0 \frac{B}{A}. \quad (3.32)$$

### 3.3.2 Circuito RLC série

Analogamente, definiremos os operadores de 1ª e 2ª ordem, em termos da corrente, de acordo com o método de diferenças finitas.

$$\frac{d^2I}{dt^2} = \frac{I(n+1) - 2I(n) + I(n-1)}{\Delta t^2}, \quad (3.33)$$

$$\frac{dI}{dt} = \frac{I(n+1) - I(n)}{\Delta t}. \quad (3.34)$$

Com  $\Delta t = h$ , reescrevemos a equação diferencial do circuito RLC série da seguinte forma,

$$\frac{I(n+1) - 2I(n) + I(n-1)}{h^2} + \frac{R}{L} \frac{I(n+1) - I(n)}{h} + \frac{1}{LC} I(n) = 0. \quad (3.35)$$

Resolvendo a equação (3.35) em termos de  $I(n+1)$ ,

$$I(n+1) = \frac{2I(n) - I(n-1)}{Ah^2} + I(n) \frac{B}{A}, \quad (3.36)$$

onde realizamos a seguinte simplificação,

$$A = \left( \frac{1}{h^2} + \frac{R}{Lh} \right), B = \left( \frac{R}{Lh} - \frac{1}{LC} \right). \quad (3.37)$$

Para resolver o problema de contorno da (3.36) usaremos a condição inicial que nos permite encontrar a resposta natural do circuito RLC série,

$$\frac{di(0)}{dt} = \frac{V_0}{L}, \quad (3.38)$$

ou em termos de diferenças finitas

$$\frac{I(n) - I(n-1)}{h} = \frac{V_0}{L}, \quad (3.39)$$

encontramos, assim, a condição que resolve o problema computacional quando fazemos ( $n=0$ ).

$$I(n-1) = I(n) - \frac{V_0}{L}. \quad (3.40)$$

Substituindo a equação (3.40) na equação (3.36), encontramos o termo geral,

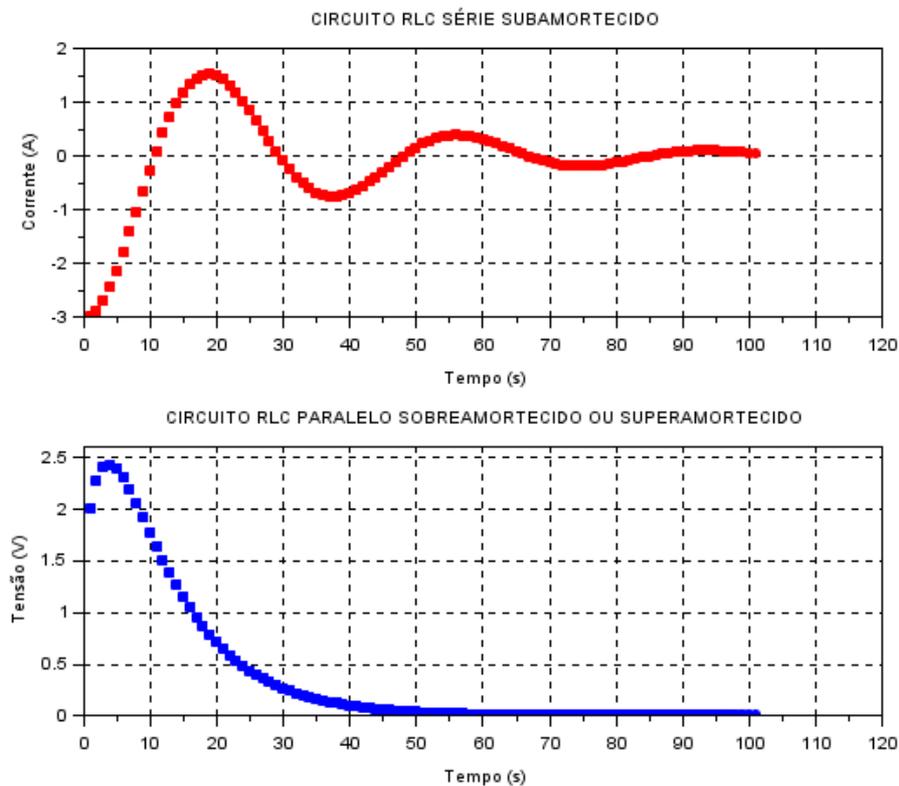
$$I(n+1) = \frac{1}{Ah^2} \left( I(n) + h \frac{V_0}{L} \right) + I(n) \frac{B}{A}, \quad (3.41)$$

a condição inicial que garante a estabilidade da simulação do circuito RLC série,

$$I(1) = \frac{1}{Ah^2} \left( I_0 + h \frac{V_0}{L} \right) + I_0 \frac{B}{A}. \quad (3.42)$$

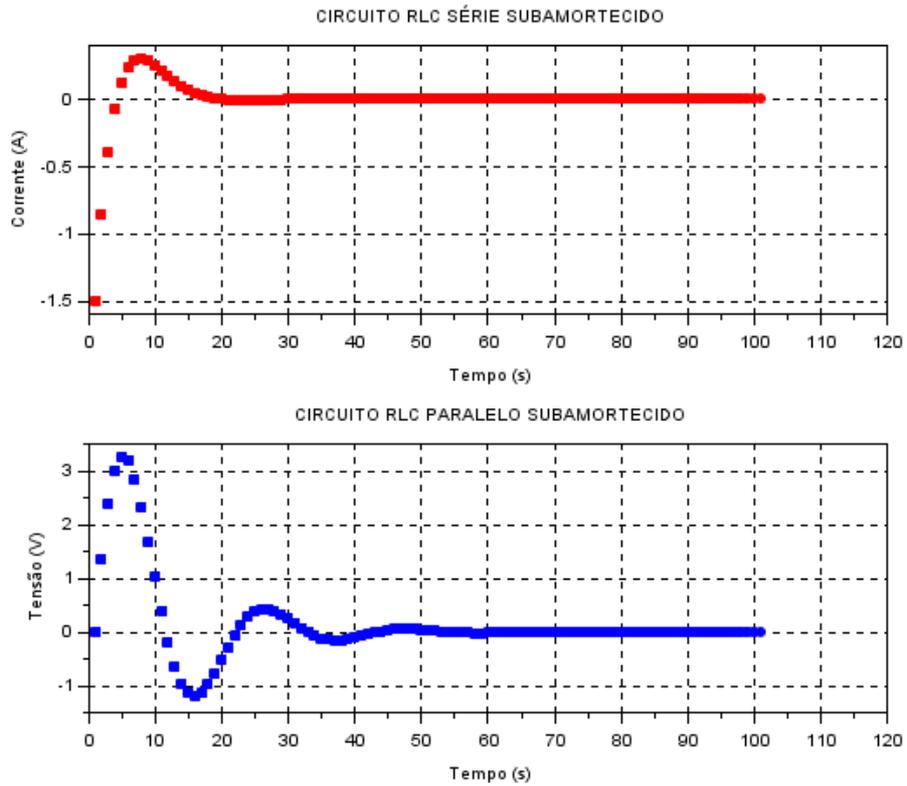
Os gráficos das figuras (18), (19) e (20) foram obtidos com o simulador de circuitos elétricos, desenvolvido com base no método das diferenças finitas.

Figura 18: Gráfico do circuito RLC Série (Caso Subamortecido) e RLC Paralelo (Caso Superamortecido), utilizando o método de diferenças finitas.



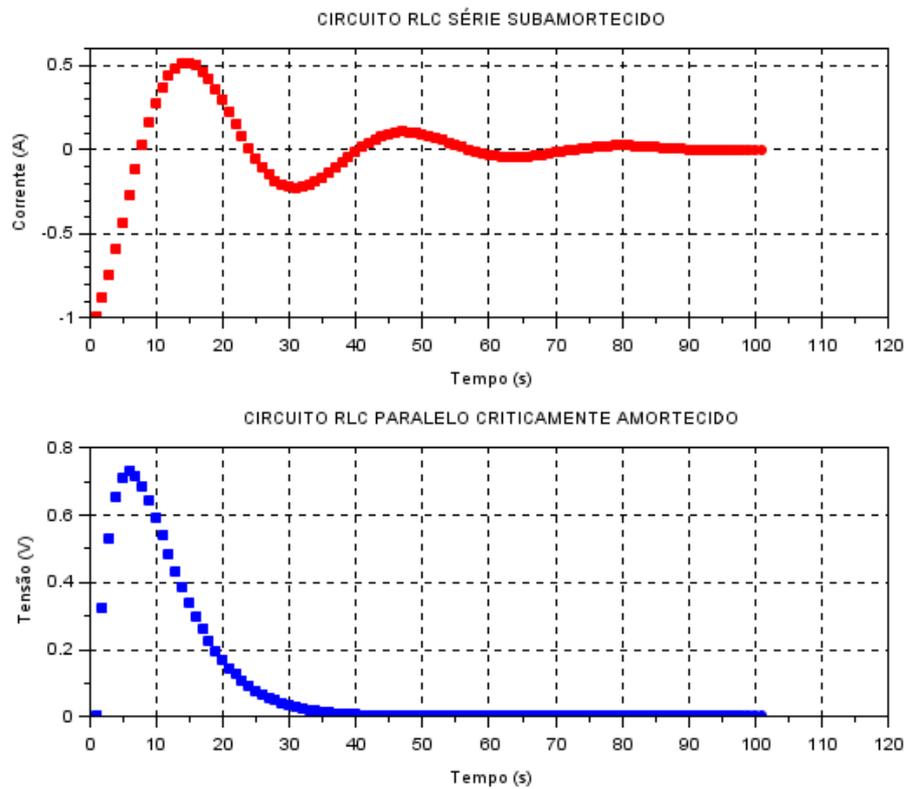
Com os seguintes valores,  $R=1\Omega$ ,  $L=4/3H$ ,  $C=1/4F$ ,  $I_0=-3A$  e  $V_0=2V$ . Fonte: Do próprio autor.

Figura 19: Gráfico do circuito RLC Série (Caso Subamortecido) e RLC Paralelo (Caso Subamortecido), utilizando o método de diferenças finitas.



Com os seguintes valores,  $R=5\Omega$ ,  $L=1H$ ,  $C=1/10F$ ,  $I_0=-3/2A$  e  $V_0=0V$ . Fonte: Do próprio autor.

Figura 20: Gráfico do circuito RLC Série (Caso Subamortecido) e RLC Paralelo (Caso Criticamente Amortecido), utilizando o método de diferenças finitas.



Com os seguintes valores,  $R=1\Omega$ ,  $L=1H$ ,  $C=1/4F$ ,  $I_0=-1A$  e  $V_0=0V$ . Fonte: Do próprio autor.

## 4 CONCLUSÃO

No trabalho apresentado utilizamos o método das diferenças finitas com o objetivo de analisar e simular os processos de carregamento e descarregamento de um capacitor em um circuito RC, e também o comportamento da resposta natural dos circuitos elétricos RLC paralelo e RLC série, todos no decorrer do tempo. Inicialmente, foi desenvolvido o programa no software Scilab, para realizar todo o processo desse trabalho, para realizar uma comparação entre a solução exata e a solução numérica obtida pelo método das diferenças finitas dos circuitos RC e RLC, com a finalidade de verificar a eficiência do método utilizado. Com o decorrer do trabalho, podemos concluir que:

- As soluções obtidas pelo método das diferenças finitas apresentaram um resultado confiável quando comparado com as soluções obtidas pelos métodos analíticos.
- A solução numérica apresentou uma mínima divergência, em relação aos valores exatos. Essa divergência pode ter sido ocasionada por diversos fatores, como por exemplo: erros originados pelo truncamento da expansão de Taylor ou pelo arredondamento realizado no computador. O simulador é capaz de mostrar os pontos, e isso permite uma rápida verificação.

Foram mostradas as ideias gerais para descrever os circuitos RC, para os processos de carregamento e descarregamento, e RLC onde, a partir das leis físicas que regem cada componente, obtemos equações diferenciais ordinárias (EDO) que descrevem o comportamento do circuito. Para o circuito RC, no processo de carregamento, consideramos o capacitor totalmente descarregado, a partir daí, foram desenvolvidas as equações da tensão e da corrente no capacitor. Já no processo de descarregamento, foi considerado o capacitor totalmente carregado, então, tomando a mesma ideia do carregamento, foram desenvolvidas as equações da tensão e da corrente no capacitor. No RLC paralelo, a tensão é comum a todos os elementos, por isso resolvemos a EDO para a tensão, enquanto no RLC série a corrente é comum a todos os elementos, por isso obtemos a resposta em termos da corrente. Duas estratégias de solução foram discutidas, a analítica que produz respostas exatas e a numérica que produz respostas aproximadas. Conforme mostram os resultados, a resposta da tensão e da corrente no circuito RLC paralelo e série, respectivamente, apresentam respostas semelhantes.

Note que existe uma diferença em resolver circuitos RLC usando a modelagem analítica e a modelagem numérica. Na primeira precisamos saber a natureza do circuito,

superamortecida, subamortecida ou amortecimento crítico, para depois montar a solução. Na segunda, a mesma solução é aplicada a todos os casos, dependendo dos valores de entrada dos parâmetros do circuito a saída pode ser um dos três casos citados.

## REFERÊNCIAS

BARRETO, L. S. **Iniciação ao Scilab**, Edição do autor, Costa de Caparica, 2008.

BERTOLDI, O. J. **Investigação de Estratégias de Otimização de Plantas Virtuais usando os Softwares Coco, Scilab e Excel**. 2012. Dissertação (Mestrado) – Programa de Pós-graduação em Engenharia Química, Faculdade de Engenharia Química. Uberlândia, 2012.

FÍSICA COMPUTACIONAL - **PROFESSOR DA USP EXPLICA**. Disponível em:<  
<https://www.youtube.com/watch?v=OQwzI3YAvHo>>. Acesso em: 16 de Junho de 2018.

FÍSICA COMPUTACIONAL. Disponível em:< <https://www.infoescola.com/fisica/fisica-computacional/>>. Acesso em: 16 de Junho de 2018.

FÍSICA COMPUTACIONAL: **A FÍSICA DO PRESENTE (E DO FUTURO)**. Disponível em:< <https://universoracionalista.org/fisica-computacional-a-fisica-do-presente-e-do-futuro/>>. Acesso em: 16 de Junho de 2018.

FÍSICA E A TECNOLOGIA. Disponível em:<<https://educador.brasilecola.uol.com.br/estrategias-ensino/fisica-tecnologia.htm>. . Acesso em: 16 de Junho de 2018.

FRANCO, N. B. **Cálculo numérico**. São Paulo: Person Prentice Hall, 2006.

History: **The 80's, from Blaise to Basile**. Disponível em:<<https://www.scilab.org/en/scilab/history>>. Acesso em 12 de Junho de 2018. Acesso em: 16 de Junho de 2018.

[http://producao.virtual.ufpb.br/books/reamat/CalculoNumerico/cap\\_scilab/cap\\_scilab.tex](http://producao.virtual.ufpb.br/books/reamat/CalculoNumerico/cap_scilab/cap_scilab.tex). Acesso em: 16 de Junho de 2018.

<https://camilasoares.wordpress.com/2009/04/07/deducao-das-equacoes-de-carga-e-descarga-dos-capacitores-utilizando-equacoes-diferenciais-de-primeira-ordem/>. Acesso em: 16 de Junho de 2018.

Johnson, David E., John L. Hilburn, and Johnny R. Johnson. **Fundamentos de análise de circuitos elétricos**. Livros Tecnicos e Cientificos, 1994.

RUGGIERO, M. A. G.; LOPES, V. L. R. **Cálculo numérico**: aspectos teóricos e computacionais. 2 ed. São Paulo: Pearson Makron Books, 1996.

## APÊNDICE A – (SIMULADOR) CÓDIGO DO PROGRAMA NO SCILAB

Produto Didático: Código, escrito na versão 6.0.1 do Scilab, para simulação dos processos de carregamento e descarregamento do capacitor em um circuito RC e da resposta natural em um circuito RLC, usando o método numérico das diferenças finitas.

```

clear;
clc;

// STRING (VALORES DAS VARIÁVEIS DE COMANDO DO SIMULADOR)
s=1;
n=2;
c=3;
d=4;
rlc=5;
rc=6;

// ENTRADA DO PROGRAMA (ÍNICIO)
mprintf("Programa para gerar sequências de pontos do processo de simulação do Circuito
RLC/RC.\n\n");

//Condição para a visualização do Circuito RLC ou RC
V=input("Deseja ver a análise do Circuito RLC ou RC? rlc/rc: ")
if V==5 then
mprintf(" #CIRCUITO RLC# \n\n")
disp('Entrada de Dados');

Resistência=input("Digite o valor da Resistência ( $\Omega$ ): R = ");
mprintf("A Resistência pode ser fracionária ou não!");
PR=input("Digite o valor do expoente da potência de 10 da Resistência: ");

Indutância=input("Digite o valor da Indutância (H): L = ");
mprintf("A Indutância pode ser fracionária ou não!");
PI=input("Digite o valor do expoente da potência de 10 da Indutância: ");

Capacitância=input("Digite o valor da Capacitância (F): C = ");
mprintf("A Capacitância pode ser fracionária ou não!");
PC=input("Digite o valor do expoente da potência de 10 da Capacitância: ");

Corrente=input("Digite o valor da Corrente inicial(A): I0 = ");
mprintf("A Corrente pode ser fracionária ou não!");
PCo=input("Digite o valor do expoente da potência de 10 da Corrente: ");

Tensão=input("Digite o valor inicial da tensão (V): V0 = ");
mprintf("A Tensão pode ser fracionária ou não!");
PV=input("Digite o valor do expoente da potência de 10 da Tensão: ");

```

```

EP=input("Digite o valor da Tensão da Bateria (V): V = ");
mprintf("A Tensão da Bateria pode ser fracionária ou não!");
PEP=input("Digite o valor do expoente da potência de 10 da Tensão da Bateria: ");

//Incremento
h=0.1;

//Resistência
R=Resistência*10^(PR);

//Indutância
L=Indutância*10^(PI);

//Capacitância
C=Capacitância*10^(PC);

//Tensão da Bateria
epsilon=EP*10^(PEP);

//Corrente inicial
I0=Corrente*10^(PCo);

//Tensão inicial
V0=Tensão*10^(PV);

//valores constantes
a=((2*C*L+(h^2))*R-L*h)/((2*C*L+(h^2))*R+L*h);
b=(4*L*C*R)/((2*C*L+(h^2))*R+L*h);
ca=V0+(((R*I0+V0)*h)/(R*C))+((1/2)*h^2*(((R*I0+V0)/(R^2*C^2))-((V0/L*C))));
a1=(-R*h+2*L)*C+h^2/((R*h+2*L)*C+h^2);
b1=(4*L*C)/((R*h+2*L)*C+h^2);
c1=I0-h*(-(V0+R*I0)/L)+(1/2)*h^2*((R*(I0*R+V0)/L^2)-(I0/(L*C)));

//condição para circuito rlc em série
alpha1=(R/(2*L));
W01=(1/(sqrt(L*C)));

//condição para circuito rlc em paralelo
alpha=(1/(2*R*C));
W0=(1/(sqrt(L*C)));

// Quantidade de pontos para o usuário escolher
N=input("Deseja simular quantos pontos? ");
mprintf('alpha1 = %.2f, W01 = %.2f\n',alpha1,W01);
mprintf('alpha = %.2f, W0 = %.2f\n',alpha,W0);

if (alpha1==W01) then
mprintf("De acordo com os parâmetros escolhidos, o circuito RLC Série é criticamente
amortecimento! (alpha1=W01)\n")
elseif (alpha1>W01)

```

```

mprintf("De acordo com os parâmetros escolhidos, o circuito RLC Série é superamortecido!
(alpha1>W01)\n")
else
mprintf("De acordo com os parâmetros escolhidos, o circuito RLC Série é subamortecido!
(alpha1<W01)\n")
end
if (alpha==W0) then
mprintf("De acordo com os parâmetros escolhidos, o circuito RLC PARALELO é
criticamente amortecimento! (alpha=W0)\n\n")
elseif (alpha>W0)
mprintf("De acordo com os parâmetros escolhidos, o circuito RLC PARALELO é
superamortecido! (alpha>W0)\n\n")
else
mprintf("De acordo com os parâmetros escolhidos, o circuito RLC PARALELO é
subamortecido! (alpha<W0)\n\n")
end
Gr=input("Deseja ver o gráfico dos circuitos RLC Série e RLC PARALELO? s/n: ");
if Gr==1 then

//Laço que gera os pontos do gráfico
for i=1:N

//condição inicial para a tensão
V(1)=V0;
v(2)=V0;

//equação da tensão inicial para o circuito RLC
v(1) = ca;

//equação da corrente inicial para o circuito RLC serie
j(1)=c1;
J(1)=I0;
j(2)=I0;

//equação da tensão para o circuito RLC em paralelo
V(i+1) = -(a*v(i)-b*V(i));
v(i+1) = V(i);

//equação da corrente para o circuito RLC em serie
J(i+1) = -(a1*j(i)-b1*J(i));
j(i+1) = J(i);

//PLOT PARA CIRCUITO RLC SÉRIE E CIRCUITO RLC PARALELO
if (alpha1==W01) then

//Gráfico da corrente
subplot(211)
plot(J,".b");
xtitle('CIRCUITO RLC SÉRIE CRITICAMENTE AMORTECIDO')

```

```

xgrid
ylabel("Corrente (A)")
xlabel("Tempo (s)")

elseif (alpha>W0)

//Gráfico da corrente
subplot(211)
plot(J,".b");
xlabel('CIRCUITO RLC SÉRIE SOBREAMORTECIDO OU SUPERAMORTECIDO')
xgrid
ylabel("Corrente (A)")
xlabel("Tempo (s)")

else

//Gráfico da corrente
subplot(211)
plot(J,".b");
xlabel('CIRCUITO RLC SÉRIE SUBAMORTECIDO')
xgrid
ylabel("Corrente (A)")
xlabel("Tempo (s)")
end
if (alpha==W0) then

//Gráfico da tensão
subplot(212)
plot(V,".r");
xlabel('CIRCUITO RLC PARALELO CRITICAMENTE AMORTECIDO')
xgrid
ylabel("Tensão (V)")
xlabel("Tempo (s)")

elseif (alpha>W0);

//Gráfico da tensão
subplot(212)
plot(V,".r");
xlabel('CIRCUITO RLC PARALELO SOBREAMORTECIDO OU SUPERAMORTECIDO')
xgrid
ylabel("Tensão (V)")
xlabel("Tempo (s)")

else

//Gráfico da tensão
subplot(212)
plot(V,".r");
xlabel('CIRCUITO RLC PARALELO SUBAMORTECIDO')

```

```

xgrid
ylabel("Tensão (V)")
xlabel("Tempo (s)")
end
end
else
mprintf("Reinicie o programa!");
end
else V==6

//Entrada de Contorno para o Processo de simulação do Circuito RC!
mprintf(" #CIRCUITO RC# \n\n")

//ENTRADA DE DADOS
disp('Entrada de Dados');
Resistência=input("Digite o valor da Resistência (Ω): R = ");
mprintf("A Resistência pode ser fracionária ou não!");
PR=input("Digite o valor do expoente da potência de 10 da Resistência: ");

Capacitância=input("Digite o valor da Capacitância (F): C = ");
mprintf("A Capacitância pode ser fracionária ou não!");
PC=input("Digite o valor do expoente da potência de 10 da Capacitância: ");

EP=input("Digite o valor da Tensão da Bateria (V): V = ");
mprintf("A Tensão da Bateria pode ser fracionária ou não!");
PEP=input("Digite o valor do expoente da potência de 10 da Tensão da Bateria: ");

//Incremento
h=0.1;

//Resistência
R=Resistência*10^(PR);

//Capacitância
C=Capacitância*10^(PC);

//Tensão da Bateria
epsilon=EP*10^(PEP);

//valores constantes
Arc=h*epsilon/R;
Brc=h*(1/(R*C)-1/h);

//Condição de Contorno para a Carga no Processo de Carregamento do Capacitor
Q(1)=0;

//Condição de Contorno para a Carga no Processo de Descarregamento do Capacitor
Q1(1)=C*epsilon;

//Condição de Contorno para a corrente inicial

```

```
I(1)=0;
```

```
//Condição de para determinar o numero de pontos na simulação
```

```
N=input("Deseja ver quantos pontos na simulação?");
```

```
//Condição de para simular os dois processos
```

```
T=input("Deseja simular os dois processos simultaneamente? s/n:");
```

```
if T==1 then
```

```
  mprintf("Gráfico dos dois Processos");
```

```
  x = input("Deseja ver o gráfico dos dois processos? s/n:");
```

```
  if x == 1 then
```

```
    // Laço FOR para gerar a sequência de pontos do processo de carregamento do Capacitor
```

```
    for t=1:N
```

```
      plot(0,epsilon*C, '.',style=1)
```

```
      plot(0,0, '.',style=1)
```

```
      plot(N,0, '.',style=1)
```

```
    //equação do carregamento do capacitor
```

```
    Q(t+1)=Arc-Q(t)*Brc;
```

```
    // Comando que mostra os pontos gerados
```

```
    disp([t,Q(t)])
```

```
    // Laço FOR para gerar a sequência de pontos do processo de descarregamento do Capacitor
```

```
    plot(0,epsilon*C, '.',style=1)
```

```
    plot(0,0, '.',style=1)
```

```
    plot(N,0, '.',style=1)
```

```
    //equação do descarregamento do capacitor
```

```
    Q1(t+1) =0*Arc-Q1(t)*Brc;
```

```
    // Comando que mostra os pontos gerados
```

```
    disp([t,Q1(t)])
```

```
    //comando para o plot da carga
```

```
    subplot(311)
```

```
    plot(Q1, '.r')
```

```
    plot(Q, '.b')
```

```
    xgrid
```

```
    xtitle('GRÁFICO DA CARGA NO CARREGAMENTO/DESCARREGAMENTO')
```

```
    ylabel("Carga (Q)")
```

```
    xlabel("Tempo (s)")
```

```
    //comando para o plot da tensão
```

```
    subplot(312)
```

```
    plot(Q1/C, '.r')
```

```
    plot(Q/C, '.b')
```

```
    xgrid
```

```
    xtitle('GRÁFICO DA TENSÃO NO CARREGAMENTO/DESCARREGAMENTO')
```

```
    ylabel("Tensão (V)")
```

```

xlabel("Tempo (s)")

//comando para o plot da corrente
subplot(313)
plot(-Q1/(R*C),'.r')
plot(-Q/(R*C),'.b')
xgrid
xtitle('GRÁFICO DA CORRENTE NO CARREGAMENTO/DESCARREGAMENTO')
ylabel("Corrente (A)")
xlabel("Tempo (s)")
end

//Gráfico do descarregamento; pontos vermelhos (".r")
// plot(Q1, '.r')
xgrid

//Gráfico do carregamento; pontos azuis (".b")
// plot(Q, '.b')
xgrid
xtitle('GRÁFICO DO DESCARREGAMENTO/CARREGAMENTO')

else x == 2
mprintf("Reinicie o programa!")
end
else T==2
P=input("Qual o processo que você deseja ver? d(Descarregamento)/c(Carregamento):");
if P==4 then
mprintf("Gráfico do Descarregamento");
x = input("Deseja ver o gráfico do descarregamento? s/n:");
if x == 1 then

// Laço FOR para gerar a sequência de pontos do processo de descarregamento do Capacitor
for t=1:N
plot(0,epsilon*C, '.',style=1)
plot(0,0, '.',style=1)
plot(N,0, '.',style=1)

Q1(t+1)=0*Arc-Q1(t)*Brc;

// Comando que mostra os pontos gerados
disp([t,Q1(t)])

//comando para o plot da carga
subplot(311)
plot(Q1, '.r')
xgrid
xtitle('GRÁFICO DA CARGA NO DESCARREGAMENTO')
ylabel("Carga (Q)")
xlabel("Tempo (s)")

```

```

//comando para o plot da tensão
subplot(312)
plot(Q1/C,'.b')
xgrid
xtitle('GRÁFICO DA TENSÃO NO DESCARREGAMENTO')
ylabel("Tensão (V)")
xlabel("Tempo (s)")

//comando para o plot da corrente
subplot(313)
plot(-Q1/(R*C),'g')
xgrid
xtitle('GRÁFICO DA CORRENTE NO DESCARREGAMENTO')
ylabel("Corrente (A)")
xlabel("Tempo (s)")
end
else
mprintf("Reinicie o programa!")
end
else P==3
mprintf("Gráfico do Carregamento")
x = input("Deseja ver o gráfico do carregamento? s/n:")
if x == 1 then

// Laço FOR para gerar a sequência de pontos do processo de carregamento do Capacitor
for t=1:N
plot(0,epsilon*C,'.',style=1)
plot(0,0,'.',style=1)
plot(N,0,'.',style=1)

Q(t+1)=Arc-Q(t)*Brc;

//comando de visualização dos pontos
disp([t,Q(t)])

//comando para o plot da carga
subplot(311)
plot(Q,'.r')
xgrid
xtitle('GRÁFICO DA CARGA NO CARREGAMENTO')
ylabel("Carga (Q)")
xlabel("Tempo (s)")

//comando para o plot da tensão
subplot(312)
plot(Q/C,'.b')
xgrid
xtitle('GRÁFICO DA TENSÃO NO CARREGAMENTO')
ylabel("Tensão (V)")
xlabel("Tempo (s)")

```

```

//comando para o plot da corrente
subplot(313)
plot(-Q/(R*C),'g')
xgrid
xtitle('GRÁFICO DA CORRENTE NO CARREGAMENTO')
ylabel("Corrente (A)")
xlabel("Tempo (s)")
end
else x == 2
mprintf("Reinicie o programa!")
end
end
end
end
M = input("Deseja simular outros procedimentos novamente? s/n: ")

// Criação de um loop utilizando while;
while M == 1
clc;

// STRING (VALORES DAS VARIÁVEIS DE COMANDO DO SIMULADOR)
s=1;
n=2;
c=3;
d=4;
rlc=5;
rc=6;

// ENTRADA DO PROGRAMA (ÍNICIO)
mprintf("Programa para gerar sequências de pontos do processo de simulação do Circuito RLC/RC.\n\n");

//Condição para a visualização do Circuito RLC ou RC
V=input("Deseja ver a análise do Circuito RLC ou RC? rlc/rc: ")
if V==5 then
mprintf(" #CIRCUITO RLC# \n\n")
disp('Entrada de Dados');
Resistência=input("Digite o valor da Resistência (Ω): R = ");
mprintf("A Resistência pode ser fracionária ou não!");
PR=input("Digite o valor do expoente da potência de 10 da Resistência: ");

Indutância=input("Digite o valor da Indutância (H): L = ");
mprintf("A Indutância pode ser fracionária ou não!");
PI=input("Digite o valor do expoente da potência de 10 da Indutância: ");

Capacitância=input("Digite o valor da Capacitância (F): C = ");
mprintf("A Capacitância pode ser fracionária ou não!");
PC=input("Digite o valor do expoente da potência de 10 da Capacitância: ");

```

```

Corrente=input("Digite o valor da Corrente inicial(A): I0 = ");
mprintf("A Corrente pode ser fracionária ou não!");
PCo=input("Digite o valor do expoente da potência de 10 da Corrente: ");

Tensão=input("Digite o valor inicial da tensão (V): V0 = ");
mprintf("A Tensão pode ser fracionária ou não!");
PV=input("Digite o valor do expoente da potência de 10 da Tensão: ");

EP=input("Digite o valor da Tensão da Bateria (V): V = "); ;
mprintf("A Tensão da Bateria pode ser fracionária ou não!");
PEP=input("Digite o valor do expoente da potência de 10 da Tensão da Bateria: ");

//Incremento
h=0.1;

//Resistência
R=Resistência*10^(PR);

//Indutância
L=Indutância*10^(PI);

//Capacitância
C=Capacitância*10^(PC);

//Tensão da Bateria
epsilon=EP*10^(PEP);

//Corrente inicial
I0=Corrente*10^(PCo);

//Tensão inicial
V0=Tensão*10^(PV);

//valores constantes
a=((2*C*L+(h^2))*R-L*h)/((2*C*L+(h^2))*R+L*h);
b=(4*L*C*R)/((2*C*L+(h^2))*R+L*h);
ca=V0+(((R*I0+V0)*h)/(R*C))+((1/2)*h^2*(((R*I0+V0)/(R^2*C^2))-((V0/L*C))));
a1=(-R*h+2*L)*C+h^2/((R*h+2*L)*C+h^2);
b1=(4*L*C)/((R*h+2*L)*C+h^2);
c1=I0-h*(-(V0+R*I0)/L)+(1/2)*h^2*((R*(I0*R+V0)/L^2)-(I0/(L*C)));

//condição para circuito rlc em série
alpha1=(R/(2*L));
W01=(1/(sqrt(L*C)));

//condição para circuito rlc em paralelo
alpha=(1/(2*R*C));
W0=(1/(sqrt(L*C)));

// Quantidade de pontos para o usuário escolher

```

```

N=input("Deseja simular quantos pontos? ");
mprintf('alpha1 = %.2f, W01 = %.2fn',alpha1,W01);
mprintf('alpha = %.2f, W0 = %.2fn\n',alpha,W0);
if (alpha1==W01) then
mprintf("De acordo com os parâmetros escolhidos, o circuito RLC Série é criticamente
amortecimento! (alpha1=W01)\n")
elseif (alpha1>W01)
mprintf("De acordo com os parâmetros escolhidos, o circuito RLC Série é superamortecido!
(alpha1>W01)\n")
else
mprintf("De acordo com os parâmetros escolhidos, o circuito RLC Série é subamortecido!
(alpha1<W01)\n")
end
if (alpha==W0) then
mprintf("De acordo com os parâmetros escolhidos, o circuito RLC PARALELO é
criticamente amortecimento! (alpha=W0)\n\n")
elseif (alpha>W0)
mprintf("De acordo com os parâmetros escolhidos, o circuito RLC PARALELO é
superamortecido! (alpha>W0)\n\n")
else
mprintf("De acordo com os parâmetros escolhidos, o circuito RLC PARALELO é
Subamortecido! (alpha<W0)\n\n")
end
Gr=input("Deseja ver o gráfico dos circuitos RLC Série e RLC PARALELO? s/n: ");
if Gr==1 then

//Laço que gera os pontos do gráfico
for i=1:N

//condição inicial para a tensão
V(1)=V0;
v(2)=V0;

//equação da tensão inicial para o circuito RLC
v(1) = ca;

//equação da corrente inicial para o circuito RLC serie
j(1)=c1;
J(1)=I0;
j(2)=I0;

//equação da tensão para o circuito RLC em paralelo
V(i+1) = -(a*v(i)-b*V(i));
v(i+1) = V(i);

//equação da corrente para o circuito RLC em serie
J(i+1) = -(a1*j(i)-b1*J(i));
j(i+1) = J(i);

```

```
//PLOT PARA CIRCUITO RLC SÉRIE E CIRCUITO RLC PARALELO
```

```
if (alpha1==W01) then
```

```
  //Gráfico da corrente
```

```
  subplot(211)
```

```
  plot(J, ".b");
```

```
  xtitle('CIRCUITO RLC SÉRIE CRITICAMENTE AMORTECIDO')
```

```
  xgrid
```

```
  ylabel("Corrente (A)")
```

```
  xlabel("Tempo (s)")
```

```
elseif (alpha1>W01)
```

```
  //Gráfico da corrente
```

```
  subplot(211)
```

```
  plot(J, ".b");
```

```
  xtitle('CIRCUITO RLC SÉRIE SOBREAMORTECIDO OU SUPERAMORTECIDO')
```

```
  xgrid
```

```
  ylabel("Corrente (A)")
```

```
  xlabel("Tempo (s)")
```

```
else
```

```
  //Gráfico da corrente
```

```
  subplot(211)
```

```
  plot(J, ".b");
```

```
  xtitle('CIRCUITO RLC SÉRIE SUBAMORTECIDO')
```

```
  xgrid
```

```
  ylabel("Corrente (A)")
```

```
  xlabel("Tempo (s)")
```

```
end
```

```
if (alpha==W0) then
```

```
  //Gráfico da tensão
```

```
  subplot(212)
```

```
  plot(V, ".r");
```

```
  xtitle('CIRCUITO RLC PARALELO CRITICAMENTE AMORTECIDO')
```

```
  xgrid
```

```
  ylabel("Tensão (V)")
```

```
  xlabel("Tempo (s)")
```

```
elseif (alpha>W0);
```

```
  //Gráfico da tensão
```

```
  subplot(212)
```

```
  plot(V, ".r");
```

```
  xtitle('CIRCUITO RLC PARALELO SOBREAMORTECIDO OU SUPERAMORTECIDO')
```

```
  xgrid
```

```
  ylabel("Tensão (V)")
```

```

xlabel("Tempo (s)")

else

//Gráfico da tensão
subplot(212)
plot(V, ".r");
xtitle('CIRCUITO RLC PARALELO SUBAMORTECIDO')
xgrid
ylabel("Tensão (V)")
xlabel("Tempo (s)")

end
end
else

mprintf("Reinicie o programa!");
end
else V==6
//Entrada de Contorno para o Processo de simulação do Circuito RC!
mprintf(" #CIRCUITO RC# \n\n")

//ENTRADA DE DADOS
disp('Entrada de Dados');
Resistência=input("Digite o valor da Resistência (Ω): R = ");
mprintf("A Resistência pode ser fracionária ou não!");
PR=input("Digite o valor do expoente da potência de 10 da Resistência: ");

Capacitância=input("Digite o valor da Capacitância (F): C = ");
mprintf("A Capacitância pode ser fracionária ou não!");
PC=input("Digite o valor do expoente da potência de 10 da Capacitância: ");

EP=input("Digite o valor da Tensão da Bateria (V): V = ");
mprintf("A Tensão da Bateria pode ser fracionária ou não!");
PEP=input("Digite o valor do expoente da potência de 10 da Tensão da Bateria: ");

//Incremento
h=0.1;

//Resistência
R=Resistência*10^(PR);

//Capacitância
C=Capacitância*10^(PC);

//Tensão da Bateria
epsilon=EP*10^(PEP);

//valores constantes
Arc=h*epsilon/R;

```

```

Brc=h*(1/(R*C)-1/h);

//Condição de Contorno para a Carga no Processo de Carregamento do Capacitor
Q(1)=0;

//Condição de Contorno para a Carga no Processo de Descarregamento do Capacitor
Q1(1)=C*epsilon;

//Condição de Contorno para a corrente inicial
I(1)=0;

//Condição de para determinar o numero de pontos na simulação
N=input("Deseja ver quantos pontos na simulação?");

//Condição de para simular os dois processos
T=input("Deseja simular os dois processos simultaneamente? s/n:");
if T==1 then
mprintf("Gráfico dos dois Processos");
x = input("Deseja ver o gráfico dos dois processos? s/n:");
if x == 1 then

// Laço FOR para gerar a sequência de pontos do processo de carregamento do Capacitor
for t=1:N
plot(0,epsilon*C,'.',style=1)
plot(0,0,'.',style=1)
plot(N,0,'.',style=1)

//equação do carregamento do capacitor
Q(t+1)=Arc-Q(t)*Brc;

// Comando que mostra os pontos gerados
disp([t,Q(t)])

// Laço FOR para gerar a sequência de pontos do processo de descarregamento do Capacitor
plot(0,epsilon*C,'.',style=1)
plot(0,0,'.',style=1)
plot(N,0,'.',style=1)

//equação do descarregamento do capacitor
Q1(t+1) =0*Arc-Q1(t)*Brc;

// Comando que mostra os pontos gerados
disp([t,Q1(t)])

//comando para o plot da carga
subplot(311)
plot(Q1,'.r')
plot(Q,'.b')

xgrid

```

```

xtitle('GRÁFICO DA CARGA NO CARREGAMENTO/DESCARREGAMENTO')
ylabel("Carga (Q)")
xlabel("Tempo (s)")

//comando para o plot da tensão
subplot(312)
plot(Q1/C,'.r')
plot(Q/C,'.b')
xgrid
xtitle('GRÁFICO DA TENSÃO NO CARREGAMENTO/DESCARREGAMENTO')
ylabel("Tensão (V)")
xlabel("Tempo (s)")

//comando para o plot da corrente
subplot(313)
plot(-Q1/(R*C),'.r')
plot(-Q/(R*C),'.b')
xgrid
xtitle('GRÁFICO DA CORRENTE NO CARREGAMENTO/DESCARREGAMENTO')
ylabel("Corrente (A)")
xlabel("Tempo (s)")
end

//Gráfico do descarregamento; pontos vermelhos (".r")
// plot(Q1, '.r')
xgrid
//Gráfico do carregamento; pontos azuis (".b")
// plot(Q, '.b')
xgrid
xtitle('GRÁFICO DO DESCARREGAMENTO/CARREGAMENTO')
else x == 2
mprintf("Reinicie o programa!")
end
else T==2
P=input("Qual o processo que você deseja ver? d(Descarregamento)/c(Carregamento):");
if P==4 then
mprintf("Gráfico do Descarregamento");
x = input("Deseja ver o gráfico do descarregamento? s/n:");
if x == 1 then

// Laço FOR para gerar a sequência de pontos do processo de descarregamento do Capacitor
for t=1:N
plot(0,epsilon*C, '.',style=1)
plot(0,0, '.',style=1)
plot(N,0, '.',style=1)

Q1(t+1)=0*Arc-Q1(t)*Brc;

// Comando que mostra os pontos gerados
disp([t,Q1(t)])

```

```

//comando para o plot da carga
subplot(311)
plot(Q1,'r')
xgrid
xtitle('GRÁFICO DA CARGA NO DESCARREGAMENTO')
ylabel("Carga (Q)")
xlabel("Tempo (s)")

//comando para o plot da tensão
subplot(312)
plot(Q1/C,'b')
xgrid
xtitle('GRÁFICO DA TENSÃO NO DESCARREGAMENTO')
ylabel("Tensão (V)")
xlabel("Tempo (s)")

//comando para o plot da corrente
subplot(313)
plot(-Q1/(R*C),'g')
xgrid
xtitle('GRÁFICO DA CORRENTE NO DESCARREGAMENTO')
ylabel("Corrente (A)")
xlabel("Tempo (s)")
end
else
mprintf("Reinicie o programa!")
end
else P==3
mprintf("Gráfico do Carregamento")
x = input("Deseja ver o gráfico do carregamento? s/n:")
if x == 1 then
// Laço FOR para gerar a sequência de pontos do processo de carregamento do Capacitor
for t=1:N
plot(0,epsilon*C,'.',style=1)
plot(0,0,'.',style=1)
plot(N,0,'.',style=1)

Q(t+1)=Arc-Q(t)*Brc;

//comando de visualização dos pontos
disp([t,Q(t)])

//comando para o plot da carga
subplot(311)
plot(Q,'r')
xgrid
xtitle('GRÁFICO DA CARGA NO CARREGAMENTO')
ylabel("Carga (Q)")
xlabel("Tempo (s)")

```

```
//comando para o plot da tensão  
subplot(312)  
plot(Q/C,'.b')  
xgrid  
xtitle('GRÁFICO DA TENSÃO NO CARREGAMENTO')  
ylabel("Tensão (V)")  
xlabel("Tempo (s)")  
  
//comando para o plot da corrente  
subplot(313)  
plot(-Q/(R*C),'g')  
xgrid  
xtitle('GRÁFICO DA CORRENTE NO CARREGAMENTO')  
ylabel("Corrente (A)")  
xlabel("Tempo (s)")  
end  
else x == 2  
mprintf("Reinicie o programa!")  
end  
end  
end  
end  
end
```