Linhas de Transmissão

Rodrigo Augusto Valeretto
Engenharia Elétrica / Computação
Universidade de São Paulo
São Carlos, Brasil
rodrigovaleretto@usp.br

Leonardo Cerce Guioto
Engenharia Elétrica / Computação
Universidade de São Paulo
São Carlos, Brasil
leonardo.cerce@usp.br

João Pedro Borges de Castro
Engenharia Elétrica / Computação
Universidade de São Paulo
São Carlos, Brasil
joaocastro@usp.br

***Abstract*—Este documento é um relatório do primeiro projeto de Ondas Eletromagnéticas, cujo objetivo é implementar um modelo de visualização de tensão e corrente (transientes) em uma linha de transmissão sem perdas através do método das diferenças finitas no domínio do tempo.**

***Index Terms*—dfdt, linhas de transmissão, transiente**

I. INTRODUÇÃO

Transientes são estados intermediários de um circuito. Eles geralmente ocorrem quando:

- uma nova carga é conectada;
- um sinal não-periódico é injetado;
- o gerador é ligado.

Estes estados progridem pelo tempo até atingir a estabilidade, que é o chamado estado estacionário.

A. Equação do Telegrafista e Transientes

A partir de uma linha de transmissão (ver figura 2), pode-se explicitar as características elétricas dessa linha, permitindo assim determinar algumas equações importantes para o desenvolvimento do projeto.

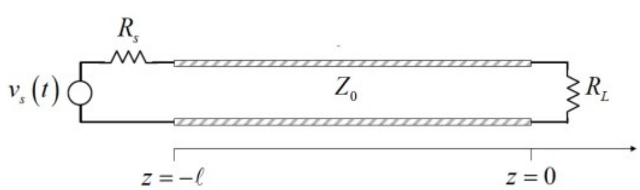

Fig. 1. Linha de transmissão.

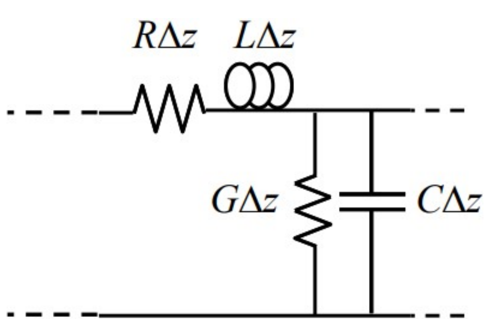

Fig. 2. Explicitação das características de um Δz da linha de transmissão.

$$-\frac{\partial v(z,t)}{\partial z} = Ri(z,t) + L\frac{\partial i(z,t)}{\partial t} \tag{1}$$

$$-\frac{\partial i(z,t)}{\partial z} = Gv(z,t) + C\frac{\partial v(z,t)}{\partial t} \tag{2}$$

As equações (1) e (2) são as chamadas equações do telegrafista, ou então equações fundamentais das linhas de transmissão.

Deriva-se (1) em relação à z e (2) em relação à t:

$$-\frac{\partial^2 v(z,t)}{\partial z^2} = R\frac{\partial i(z,t)}{\partial z} + L\frac{\partial^2 i(z,t)}{\partial t \partial z} \tag{3}$$

$$-\frac{\partial^2 i(z,t)}{\partial t \partial z} = G\frac{\partial v(z,t)}{\partial t} + C\frac{\partial^2 v(z,t)}{\partial t^2} \tag{4}$$

Substituindo (4) em (3):

$$-\frac{\partial^2 v(z,t)}{\partial z^2} = R\frac{\partial i(z,t)}{\partial z} - LG\frac{\partial v(z,t)}{\partial z} - LC\frac{\partial^2 v(z,t)}{\partial t^2} \tag{5}$$

Substituindo equações já conhecidas (1) e (2), e aplicando a condição de linha de transmissão sem perdas ($R = G = 0$):

$$\frac{\partial^2 v(z,t)}{\partial z^2} = LC\frac{\partial^2 v(z,t)}{\partial t^2} \tag{6}$$

$$\frac{\partial^2 i(z,t)}{\partial z^2} = LC\frac{\partial^2 i(z,t)}{\partial t^2} \tag{7}$$

Caso sistema atenda às equações (6) e (7), então ele pode ser utilizado para transportar informações na forma de ondas eletromagnéticas. A resolução dessas equações diferenciais mostra que temos propagação tanto para a direção positiva quanto negativa de z (para ambos v e i):

$$v(z,t) = v^+\Big(t - \frac{z}{v_f}\Big) + v^-\Big(t + \frac{z}{v_f}\Big), v_f = \frac{1}{\sqrt{LC}} \tag{8}$$

Como a velocidade de propagação é finita, dada por v_f, a tensão e a corrente na linha de transmissão passam por estados transientes antes de atingir o estado estacionário.

Denotando v_s da figura (1) em notação fasorial como V_s e invertendo o sentido do sistema de coordenadas, deriva-se a figura (3).

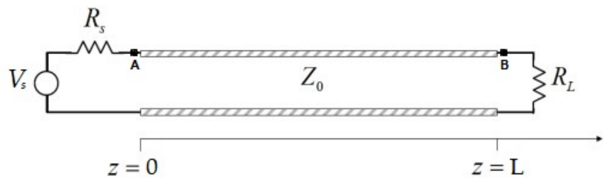

Fig. 3. Linha de transmissão para cálculo dos transientes.

É necessário, de início, definir algumas condições de contorno envolvendo os pontos A ($z = 0$) e B ($z = L$). Tem-se, em A:

$$V_s = V_A + R_s I_A \tag{9}$$

e em B:

$$V_B = R_L I_B \tag{10}$$

Alguns períodos de tempo são cruciais para entender a evolução dos transientes. São eles $t_1 = \frac{L}{v_f}$ (momento em que a onda alcança a carga e é refletida) e $t_2 = \frac{2L}{v_f}$ (momento em que a onda alcança a fonte e é refletida).

Além disso, da figura (3), temos R_S como a resistência interna da fonte de tensão, Z_0 como a impedância característica da linha de transmissão, R_L como a resistência da carga conectada nessa mesma linha e as tensões e correntes de propagação como V^+, V^-, I^+ e I^-. Assim, em $0 \leq t \leq t_1$:

$$V_s = V^+ + R_s I^+ \tag{11}$$

mas,

$$I^+ = \frac{V^+}{Z_0} \tag{12}$$

Assim, substituindo (12) em (11) e resolvendo:

$$V^+ = \frac{Z_0}{R_s + Z_0} V_s \tag{13}$$

$$I^+ = \frac{1}{R_s + Z_0} V_s \tag{14}$$

Estas são tensão e corrente na linha antes da primeira reflexão. Para $t_1 \leq t \leq t_2$:

$$V^- = \Gamma_L V^+ \tag{15}$$

$$I^- = -\Gamma_L I^+ \tag{16}$$

onde

$$\Gamma_X = \frac{R_X - Z_0}{R_X + Z_0}, X \in \{s, L\} \tag{17}$$

A tensão e corrente após a primeira reflexão são, portanto:

$$V_{ref} = V^+ + V^- = V^+(1 + \Gamma_L) \tag{18}$$

$$I_{ref} = I^+ + I^- = I^+(1 - \Gamma_L) \tag{19}$$

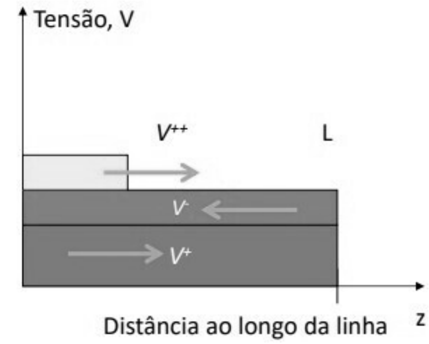

Fig. 4. Evolução dos transientes numa linha de transmissão.

Por fim, quando $t \to \infty$ (estado estacionário), tem-se:

$$V_{final} = \frac{R_L}{R_s + R_L} V_s \tag{20}$$

$$I_{final} = \frac{1}{R_s + R_L} V_s \tag{21}$$

B. *Diferenças Finitas no Domínio do Tempo (DFDT)*

O DFDT é um algoritmo de simulação que trabalha as derivadas como diferenças finitas. O sistema de equações, depois de ser simplificado, é continuamente incrementado de pequenos intervalos temporais, ultimamente calculando a solução do sistema do próximo Δt utilizando o resultado atual.

Inicialmente é necessário definir uma malha; ela determinará o conjunto finito de pontos que serão trabalhados pelo algoritmo. Considera-se z_k e t_n como os eixos. Assim:

$$z_k = k\Delta z \tag{22}$$

$$t_n = n\Delta t \tag{23}$$

com k e $n \in Z$.

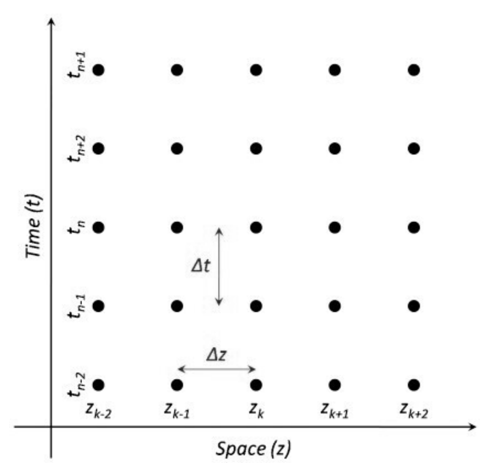

Fig. 5. Malha obtida a partir de (22) e (23).

Tendo a malha definida, o próximo passo é simplificar as equações diferenciais. Do Cálculo, tem-se que:

$$\frac{\partial v(z,t)}{\partial z} \approx \frac{v(z+\Delta z,t)-v(z-\Delta z,t)}{2\Delta z} \tag{24}$$

O mesmo pode ser escrito para a corrente. Reescrevendo $v(z_k,t_n)$ como v_k^n e aplicando as aproximações acima:

$$\frac{\partial}{\partial z}v_k^n \approx \frac{v_{k+1}^n - v_{k-1}^n}{2\Delta z} \tag{25}$$

$$\frac{\partial}{\partial t}i_k^n \approx \frac{i_k^{n+1} - i_k^{n-1}}{2\Delta t} \tag{26}$$

Embora essa relação já seja suficiente para a determinação das equações, é possível realizar uma melhoria. Definindo os pontos v_k^n da malha de forma que fiquem a uma distância de $0,5\Delta z$ (ou $0,5\Delta t$ no caso de i_k^n), a aproximação agora é dada por:

$$\frac{\partial}{\partial z}v_k^n \approx \frac{v_{k+0,5}^n - v_{k-0,5}^n}{\Delta z} \tag{27}$$

$$\frac{\partial}{\partial t}i_k^n \approx \frac{i_k^{n+0,5} - i_k^{n-0,5}}{\Delta t} \tag{28}$$

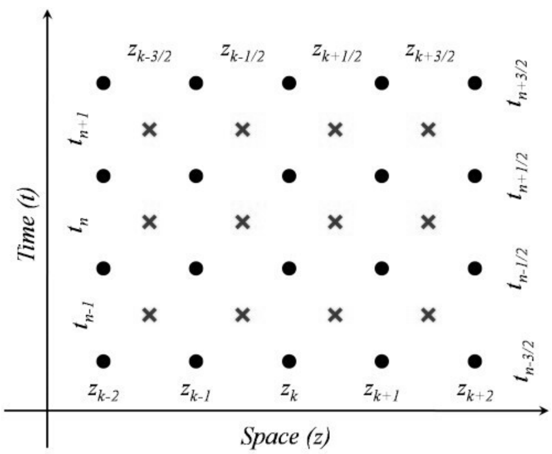

Fig. 6. Modelo final de malha a ser usada pelo algoritmo.

Essa otimização permite que o Δz ou Δt não sejam multiplicados por dois durante a aproximação, o que diminui o erro sem afetar o desempenho geral do algoritmo. Substituindo (27) e (28) em (1) e (2) e sabendo que para linhas de transmissão sem perdas $R = G = 0$:

$$-\frac{v_{k+0,5}^n - v_{k-0,5}^n}{\Delta z} = L\frac{i_k^{n+0,5} - i_k^{n-0,5}}{\Delta t} \tag{29}$$

$$-\frac{i_{k+1}^{n+0,5} - i_k^{n+0,5}}{\Delta z} = C\frac{v_{k+0,5}^{n+1} - v_{k+0,5}^n}{\Delta t} \tag{30}$$

Estas duas últimas equações são aproximações diretas das equações do telegrafista. Para determinar o próximo estado do sistema a partir dos atuais, isola-se os termos ”futuros” (que, para (29), é $i_k^{n+0,5}$, e para (30) é $v_{k+0,5}^{n+1}$):

$$i_k^{n+0,5} = -\frac{\Delta t}{L\Delta z}\left(v_{k+0,5}^n - v_{k-0,5}^n\right) + i_k^{n-0,5} \tag{31}$$

$$v_{k+0,5}^{n+1} = -\frac{\Delta t}{C\Delta z}\left(i_{k+1}^{n+0,5} - i_k^{n+0,5}\right) + v_{k+0,5}^n \tag{32}$$

Com estas equações e a partir de uma condição inicial, é possível calcular os estados do sistema de forma iterativa. No entanto, certas configurações podem se mostrar instáveis. Por isso, é necessária uma relação entre Δz e Δt:

$$\Delta t \le \frac{\Delta z}{v_f} \tag{33}$$

O limite superior de Δt é chamado de tempo crítico, a partir do qual o modelo se torna instável.

Por fim, para finalizar o algoritmo, basta definir algumas condições de contorno. Pode-se definir a condição como um curto-circuito ou então um circuito aberto. Em uma simulação com domínio de tamanho K, para a primeira situação, vale as seguinte igualdade:

$$v_{K+0,5}^{n+1} = 0 \tag{34}$$

Esta condição de contorno também é conhecida como *condição de Dirichlet*. Para a segunda:

$$v_{K+0,5}^{n+1} = v_{K-0,5}^{n+1} \tag{35}$$

Já esta segunda é chamada de *condição de Neumann*. Essas condições especiais são interessantes pois, além de serem simples de implementar, também estão relacionadas. Se, por exemplo, for escolhida a condição de $Neumann$ para a tensão, a corrente seguirá a condição de $Dirichlet$, e vice-versa.

II. Código

A. Linguagem de programação

A linguagem de programação utilizada no projeto foi a linguagem *Python*. Decidimos usar essa linguagem por ela ter inúmeras bibliotecas que podem ser importadas e utilizadas, dando-nos uma maior variedade de opções sobre como tratar o código e apresentar os resultados. Além disso, a velocidade de cálculo do *Python* foi bastante incrementada nos últimos tempos e os discentes possuiam maior afinidade com a linguagem.

B. Instalação do Python

1) Python no Linux: Se estiver em uma distribuição *Linux*, muito provavelmente a linguagem já está instalada no terminal e é possível simplesmente utilizá-la com um dos comandos:

$$python \;\; nome-do-arquivo$$
$$python3 \;\; nome-do-arquivo$$

Caso não tenha o *Python* instalado no *Linux*, basta executar o seguinte comando no terminal (distribuições Ubuntu):

$$sudo \;\; apt-get \;\; install \;\; python3$$

Isso já é o suficiente para possibilitar o uso da linguagem. Adicionalmente, será necessário a instalação do gerenciador de pacotes do *Python*. Para instalá-lo, execute o código abaixo no terminal:

$$sudo\ \ apt-get\ \ install\ \ python3-pip$$

2) Python no Windows: Para instalar o *Python* em uma máquina cujo sistema operacional é o *Windows*, acesse o site *https://www.python.org/*, navegue para a aba Downloads e baixe a versão mais recente disponível. Em seguida, basta executar o instalador e o *Python* estará integrado em sua máquina. Abra o terminal do *Windows* (*PowerShell* ou *CMD*) e execute o código com:

$$python\ \ ./nome-do-arquivo$$

No caso do *Windows*, o gerenciador de pacotes do *Python* já vem instalado, logo não é necessário instalá-lo separadamente.

C. Instalação das bibliotecas

O código desenvolvido usa algumas bibliotecas extras para poder facilitar a realização de operações matemáticas e também permitir o *plot* dos gráficos de resultado. Portanto, se faz necessário que essas bibliotecas sejam instaladas com o gerenciador de pacotes do *Python* (*pip*).

As bibliotecas usadas foram:

- numpy - usada para calculos matemáticos;
- matplotlib - usada para *plot* e animação do gráfico.

Para instalar essas bibliotecas execute os seguintes comandos em seu terminal *Linux* ou *Windows*:

$$\begin{gathered}pip3\ \ install\ \ numpy\\ pip3\ \ install\ \ matplotlib\end{gathered}$$

Caso esteja usando uma versão anterior do *Python* execute:

$$\begin{gathered}pip\ \ install\ \ numpy\\ pip\ \ install\ \ matplotlib\end{gathered}$$

É importante ressaltar que é extremamente recomendado a utilização de *Python3* para a execução do código e o funcionamento do mesmo não é garantido em versões anteriores.

Com isso é possível executar o código e observar os resultados obtidos.

D. Estrutura e explicação do código

Para a implementação do código com o algoritmo FDTD, inicialmente foi necessário definirmos os parâmetros e constantes que seriam utilizados em nossa linha de transmissão. Algumas das constantes são a velocidade da luz (c) e a velocidade de propagação na linha ($u_f = 0,9 \cdot c$).

Depois, estipulamos o tamanho da nossa linha (l_{max}). No momento estamos utilizando $10000m$ $(10km)$, porém esse valor pode ser alterado se desejado. Além disso, é necessário denotar o número de divisões na linha, o qual definimos como K e atribuímos o valor 100.

Com esses dados podemos calcular os outros termos da nossa relação, tendo em mente o critério de estabilidade de (33) e também o tempo estacionário definido como $t_{est} = 10\frac{l_{max}}{u_f}$, que representa 10 reflexões do sinal na linha. Para isso fazemos:

$$\Delta z = \frac{l_{max}}{K} \tag{36}$$

Com os valores de Δz e u_f calculamos Δt:

$$\Delta t = \frac{\Delta z}{u_f} \tag{37}$$

E, por fim, multiplicamos Δt por uma constante menor do que 1, no nosso caso $0,5$, para garantir a estabilidade, pois a igualdade pode gerar um sistema instável.

$$\Delta t = \Delta t \cdot 0,5 \tag{38}$$

Tendo Δt definido, calculamos N, o número de divisões do tempo:

$$N = int(\frac{t_{est}}{\Delta t}) \tag{39}$$

O termo int denota a necessidade de N ser um número inteiro por se tratar de um número de divisões.

Com o critério de estabilidade definido, partimos para o cálculo dos elementos utilizados ao longo da linha, como a indutância e a capacitância. Por se tratar de uma linha sem perdas, esses termos podem ser calculados a partir dos dados do problema, levando em consideração:

$$Z_0 = \sqrt{\frac{L}{C}} \tag{40}$$

Sendo L o valor da indutância e C o valor da capacitância, e

$$u_f = \frac{1}{\sqrt{LC}} \tag{41}$$

Manipulando as equações 40 e 41, encontramos:

$$L = \frac{Z_0}{u_f} \tag{42}$$

$$C = \frac{1}{Z_0 u_f} \tag{43}$$

Com esses valores calculados podemos avançar para o código usado no cômputo dos valores de $i(z,t)$ e $v(z,t)$. É importante ressaltar que utilizamos uma adaptação do código mais teórico apresentado acima, para facilitar ambos cálculo e implementação, porém evitando erros da mesma forma.

Inicialmente, instanciamos as matrizes onde vamos guardar nossos resultados, sendo elas três, uma matriz V com dimensão (N,K), uma matriz I com dimensão $(N,K-1)$ e uma matriz para a variável auxiliar Va com dimensão (N,K), lembrando que a dimensão (N,K) implica em uma matriz com índices de 0 a $N-1$ e 0 a $K-1$. Enfatizamos que a partir desse ponto a notação X_j^i será utilizada para denotar o valor de uma matriz X no ponto $n=i$ e $k=j$, ou seja é equivalente à $X[i,j]$.

Para evitar o uso de indices não inteiros $(0,5)$ e melhorar o desempenho ao livrar-nos de algumas contas, definimos um deslocamento de índices sendo $n+0,5$ como n e $k+0,5$

como k; criamos uma variável auxiliar para a tensão que denominamos Va e estipulamos:

$$Va_k^n = \frac{C\Delta z}{\Delta t} \cdot v_k^n \tag{44}$$

Com essas mudanças, já é possível aplicar o algoritmo modificado. Para começar, determinamos as condições de contorno para nosso problema, sendo que existem as condições de contorno gerais e uma específica para o caso de curto-circuito. Das gerais temos:

$$V_0^0 = \frac{Z_0}{R_s + Z_0} V_s(0) \tag{45}$$

$$I_0^0 = \frac{V_s(0)}{R_s + Z_0} \tag{46}$$

Nas equações (45) e (46) temos que V_s é a tensão da fonte escolhida para $t = 0$, Z_0 e R_s são a impedância característica da linha e a resistência interna da fonte, respectivamente, e são dados pelo enunciado do problema, assim como os valores para $V_s(t)$. Além destas condições, temos:

$$Va_k^0 = 0, \qquad k = 1, ..., K \tag{47}$$

$$I_k^0 = 0, \qquad k = 1, ..., K-1 \tag{48}$$

Para $n > 0$ fazemos:

$$Va_0^n = (1 - \beta_1) Va_0^{n-1} - 2I_0^{n-1} + \frac{2}{R_s} V_s((n-1) \cdot \Delta t) \tag{49}$$

$$Va_{K-1}^n = (1 - \beta_2) Va_{K-1}^{n-1} + 2I_{K-2}^{n-1} \tag{50}$$

E, por fim, definimos as fórmulas das iterações:

$$Va_k^n = Va_k^{n-1} - (I_k^{n-1)} - I_{k-1}^{n-1}), \qquad k = 1, ..., K-2 \tag{51}$$

$$I_k^n = I_k^{n-1} - \beta_3 (Va_{k+1}^n - Va_k^n), \qquad k = 0, ..., K-2 \tag{52}$$

Abaixo estão indicados os valores dos betas, constantes usadas nos cálculos das iterações:

$$\beta_1 = \frac{2\Delta t}{R_s C \Delta z} \tag{53}$$

$$\beta_2 = \frac{2\Delta t}{R_L C \Delta z} \tag{54}$$

$$\beta_3 = \frac{(\Delta t)^2}{LC(\Delta z)^2} \tag{55}$$

Com as definições acima podemos estruturar o código e computar os valores de $v(z,t)$ e $i(z,t)$ para os casos em que a resistência da carga tem valores inteiros diferentes de 0, inclusive o caso de circuito aberto (no qual a carga vale $R_L = \infty$); porém, para o caso em que o circuito se encontra em curto (carga $R_L = 0$), devemos fazer uma modificação na condição da equação 50 alterando-a para:

$$Va_{K-1}^n = 0 \tag{56}$$

A partir dessa última condição reestruturamos o código para o caso em que $R_L = 0$ e podemos então calcular os resultados da linha de transmissão para todos os casos requisitados pelo enunciado. Após o cômputo e preenchimento das matrizes de resultado Va e I, retornamos os resultados da variável auxiliar para a tensão padrão, realizando, portanto, o processo inverso da equação 44:

$$V_k^n = \frac{\Delta t}{C \Delta z} Va_k^n \tag{57}$$

Dessa forma obtemos o valor das matrizes V e I e realizamos o *plot* em uma animação que mostra o valor de V e I para toda a reta Z em um tempo t que varia de acordo com a animação.

O código no qual implementamos o algoritmo pode ser encontrado no diretório do github a seguir:

https://github.com/RodrigoValeretto/Trabalhos-Ondas/tree/master/Trab1-Ondas

III. Resultados e Conclusão

TABLE I
TRANSIENTES PARA $V_s = 2u(t)$ E $Z_L = 100\ \Omega$

	Intervalos			
	$t < t_1$	$t_1 < t < t_2$	$t_2 < t < t_3$	$t \to \infty$
Tensão (V)	0,800	1,067	1,120	1,143
Corrente (A)	0,016	0,0107	0,0117	0,0114

TABLE II
TRANSIENTES PARA $V_s = 2u(t)$ E $Z_L = 0\ \Omega$

	Intervalos			
	$t < t_1$	$t_1 < t < t_2$	$t_2 < t < t_3$	$t \to \infty$
Tensão (V)	0,800	0	-0,160	0
Corrente (A)	0,016	0,032	0,029	0,026

TABLE III
TRANSIENTES PARA $V_s = 2u(t)$ E $Z_L = \infty$

	Intervalos			
	$t < t_1$	$t_1 < t < t_2$	$t_2 < t < t_3$	$t \to \infty$
Tensão (V)	0,800	1,600	1,760	2,000
Corrente (A)	0,016	0	0,003	0

Analisando as tabelas I,II e III, que contém os valores teóricos esperados para as duas primeiras reflexões e para quando $t \to \infty$, obtidos através das equações de (13) a (21), e comparando com os valores da simulação, observa-se que, para o caso de $V_s = 2u(t)$, estão condizentes. É importante reforçar que pequenas variações entre os resultados são causados pela maior precisão do programa.

No entanto, se os cálculos forem feitos para o segundo valor de tensão do gerador, o mesmo não pode ser dito. Isso ocorre pois esse valor é, aproximadamente, um impulso. Logo, após uma série de reflexões, não há acúmulo de tensão e/ou corrente.

Para o caso $R_L = 100\ \Omega$, o esperado é um pulso inicial de amplitude $0,4V$, que ao refletir, assume momentaneamente a amplitude de $0,53V$ e depois decai, pois somente $0,13V$ são refletidos. No caso de $R_L = 0\ \Omega$, após a primeira reflexão, não ocorre uma diminuição de amplitude, mas o pulso é invertido.

E para $R_L = \infty$, o pulso somente é refletido, não ocorrendo inversão.

Nesses dois últimos casos, a diminuição de amplitude ocorre somente no retorno à fonte. Por fim, para todos, a amplitude é continuamente reduzida a cada reflexão até estabilizar em $0V$. O mesmo pode ser dito para a corrente.

Portanto, conclui-se que a simulação, considerando os resultados teóricos, está correta. O projeto foi bem proveitoso para os discentes. Simular algo teórico e visualizar os resultados através de uma animação é bastante recompensador; além disso, o projeto ajudou a desmistificar o funcionamento do método DFDT e das linhas de transmissão. Esperamos que os próximos projetos sejam interessantes como esse.

References

[1] D. K. Cheng, ”Field and Wave Electromagnetics”, Pequim: Tsinghua University Press, 2a. ed., 2006.

[2] M. N. O. Sadiku, ”Elementos de Eletromagnetismo”, Porto Alegre: Bookman, 3a. ed., 2004.

[3] J. R. Nagel, ”The Finite-Difference Time-Domain (FDTD) Algorithm”, acesso através do link: https://utah.instructure.com/courses/297816/assignments/1750259.

[4] H. A. Haus, J. R. Melcher, ”Eletromagnetic Field and Energy”, acesso através do link: https://web.mit.edu/6.01_book/www/.

[5] D. B. Davidson, ”Computational Eletromagnetics for RF and Microwave Engineering”, Cambridge University Press, 2005.