Instituto Tecnológico de Aeronáutica

Aplicativo web para casamento de toco simples

André Marcello Soto Riva Figueira
Daniel Prince Cerneiro
Dylan N. Sugimoto
Gabriel Adriano de Melo

São José dos Campos
23/05/2017

Introdução

Este projeto tem como objetivo a elaboração de um programa que realize os cálculos para casamento de uma linha de transmissão com uma carga pelo método do toco simples e que tenha uma interface amigável tendo em vista a grande dificuldade que é realizar esses cálculos manualmente, e o fato de existir poucos aplicativos que desempenham a mesma funcionalidade.

O objetivo final do casamento do toco simples é obter a partir da impedância intrínseca do circuito, da impedância, da frequência de casamento e das propriedades eletromagnéticas do material (permissividade elétrica e permeabilidade magnética) o comprimento do toco e a distância que esse deve ser colocado em paralelo para que não haja ondas refletidas. Para isso, primeiramente calcula-se a impedância normalizada de carga:

$$z_L = \frac{Z_L}{Z_0} \quad (1),$$

em que Z_L é a impedância de carga, Z_0 é impedância intrínseca do circuito e z_L é impedância normalizada de carga.

Em seguida, calcula-se a admitância normalizada de carga:

$$y_L = \frac{1}{z_L} \quad (2),$$

em que z_L é a impedância normalizada de carga e y_L é a admitância normalizada de carga.

Calculada a admitância de carga, substitui-se o seu valor na equação (3) e calcula-se a expressão da admitância da linha em função da posição e realiza-se a manipulação algébrica necessária para deixar a expressão na sua forma complexa algébrica.

$$y(x) = \frac{y_L + j\tan(\beta x)}{1 + y_L \tan(\beta x)} \quad (3),$$

em que y(x) é admitância da linha em função de x, que é a posição; y_L é a admitância de carga e β é número de onda.

Após o cálculo da admitância da linha, calcula-se para qual valor de posição que a parte real da admitância da linha assume valor unitário. Esse valor de posição é a posição em que o toco deve ser inserido em paralelo ao circuito. Em seguida, calcula-se o valor da parte imaginária da admitância da linha para o valor de posição encontrado anteriormente que é o valor da admitância que o toco deve subtrair para que haja o casamento; logo, a partir desse valor calcula-se o

comprimento do toco utilizando a equação (4), se o toco estiver em aberto ou a equação (5), se o toco estiver em curto. Após esses cálculos é possível calcular o V_{swr}, para outras frequências calculando, para a posição do toco, a posição elétrica correspondente para a nova frequência e com a posição elétrica calcular a nova admitância da linha usando a equação (3). Calculado a admitância da linha, soma-se esse valor ao novo valor da admitância do toco que foi calculado para o valor de comprimento elétrico do toco para a nova frequência. Com isso, calcula-se o valor do coeficiente de reflexão pela equação (6) a partir do qual se obtém, o valor do V_{swr} usando a equação (7).

$$y_{toco} = jtan(\beta l) \quad (4),$$

em que y_{toco} é a admitância do toco, l é o comprimento do toco e β é o número de onda.

$$y_{toco} = -jcotan(\beta l) \quad (5),$$

em que y_{toco} é a admitância do toco, l é o comprimento do toco e β é o número de onda.

$$|\Gamma| = \frac{|1 - y(x)|}{|1 + y(x)|} \quad (6),$$

em que Γ é o coeficiente de reflexão e y é admitância da linha em função de x.

$$V_{swr} = \frac{1 + |\Gamma|}{1 - |\Gamma|} \quad (7),$$

em que Γ é o coeficiente de reflexão e V_{swr} é taxa de voltagem da onda estacionário (Voltage Standing Wave Ratio).

Esse são os passos teóricos que o programa desse projeto automatizou na forma de um aplicativo *Web Browser* para ter uma interface mais amigável para o usuário, de forma a se tornar uma ferramenta interessante na área de ensino de engenharia eletrônica, em especial, no estudo das linhas de transmissão.

A escolha da plataforma web, sugestão inicial do nosso grupo, se deu pela acessibilidade e pela compatibilidade de tal plataforma, que pode ser acessada livremente e por qualquer sistema operacional, inclusive em tablets e smartphones que possuam navegador web.

Descrição do Algoritmo

Da linha 106 até 120 do código 3 no Anexo, os valores de inputs de entrada são transferidos para variáveis. Em seguido é calculado a impedância normalizada da carga na linha 121, e admitância normalizada da carga na linha 122. Da linha 169 até a linha 188, os cálculos necessários para se obter o valor da posição do toco, que é obtido nas linhas 189 e 190, são realizados. Após calculado a posição do toco, é calculado o comprimento do toco da linha 192 até a linha 200.

Após calculado o valor do comprimento do toco e da posição do toco, são realizados os cálculos para se obter o valor de V_{swr} para a frequência máxima e mínima definido pela banda de largura de frequência (input BW), da linha 203 até 227, em que é utilizado as funções calcularNovoyToco (linha 30 até 35), que calcula o valor da admitância para um dado valor de comprimento do toco e frequência de onda; calcularModR (linha 51 até 63), que calcula o valor do módulo do coeficiente de reflexão para um dado valor de admitância do toco, posição do toco, frequência de onda e admitância de carga; e calcularV_{swr}, que calcula o valor de V_{swr} para um dado valor de módulo do coeficiente de reflexão. Por fim, nas linhas 265 até 288, os resultados encontrados são formatados para serem exibidos na tela.

Para a exibição da carta de Smith, utiliza-se primeiro uma imagem de fundo fixa, e com posições do seu centro e raio conhecidas, e faz-se uma transformação de coordenadas para que o ponto a ser desenhado na carta seja correspondente às marcações da imagem.

Desta forma, determinou-se a transformação de coordenadas a partir do cálculo do coeficiente de reflexão, uma vez que sistemas de coordenadas para o coeficiente de reflexão é o cartesiano, com o centro na Carta de Smith, sendo apenas necessário a computação de um fator de escala para atender as dimensões da figura.

Para facilitar o seu uso para os próximos laboratórios, utilizou-se o paradigma da orientação a objeto para encapsular o código da Carta de Smith, facilitando a sua reutilização e a verificação de erros. Assim, ao criar uma nova Carta de Smith, o programador deve passar como parâmetros o centro da imagem de funda da Carta de Smith, o seu raio, a referência à imagem e a referência da área de desenho.

O algoritmo realiza sucessivas iterações para o cálculo de valores de SWR relacionados a frequências da banda de operação especificada. Para a exibição do gráfico de SWR utilizou-se a biblioteca plotly.js, que utiliza a lista de pontos de SWR gerados anteriormente.

Utilização

O aplicativo web é de fácil utilização e de grande portabilidade. Para que o usuário utilize-o, basta inserir os valores nas caixas de texto e pressionar o botão de “Calcular”. Os resultados são adicionados na página, representados pelo texto, pelas imagens da Carta de Smith (de impedância) e pelos gráficos.

Para refazer o cálculo, basta pressionar novamente o botão de “Calcular” e todos os resultados serão atualizados.

Como observado na Figura 1, o usuário deverá digitar a impedância da linha de trasmissão “Z_0”, a impedância da carga “Z_L”, com a sua parte real e imaginária. É importante notar que as unidades de impedância, apesar de serem em Ohms por padrão, podem ser qualquer unidade (desde que ambos na mesma unidade), uma vez que é a impedância normalizada da carga que será utilizada para os cálculos.

No campo de “Frequência de Casamento (MHz)”, o usuário deverá digitar a frequência para o qual o casamento será realizado, e no campo de “BW de operação (MHz)”, a largura de banda para o cálculo do SWR. Nos campos ϵ_{rel} e U_{rel}, o usuário deverá inserir a permissividade elétrica relativa e a permeabilidade magnética relativa da linha de transmissão.

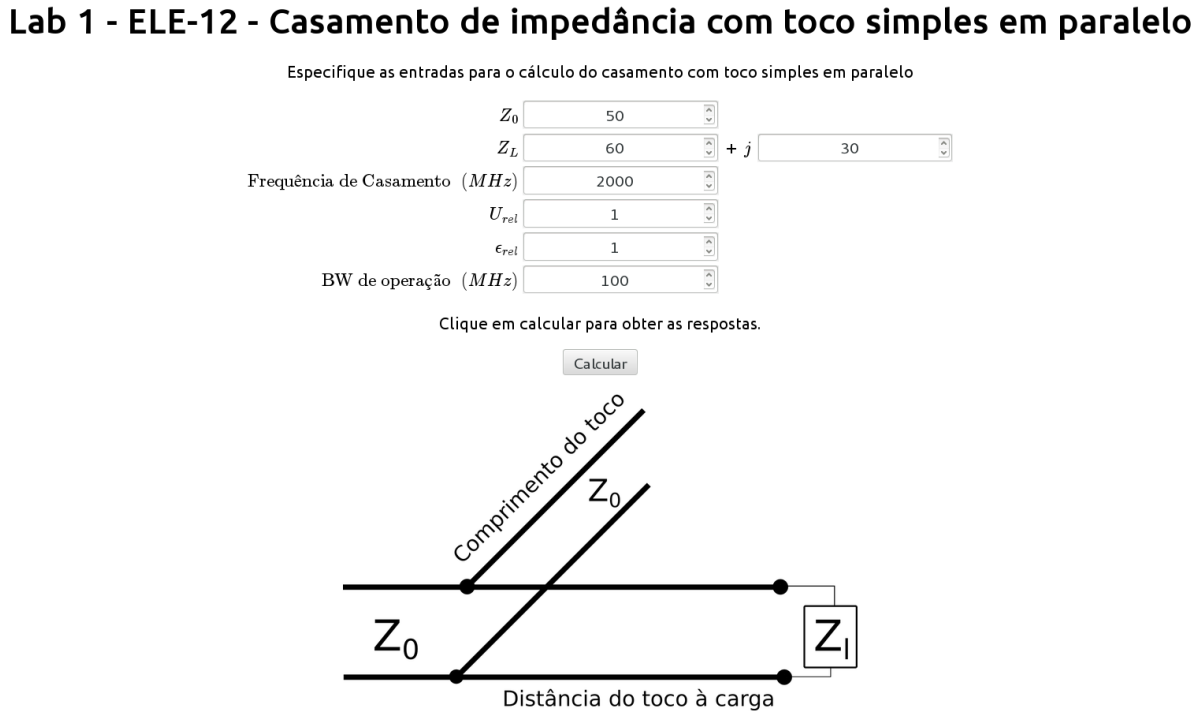

Figura 1 – Interface de entrada dos valores para o casamento.

Assim, após pressionar o botão de calcular, os resultados serão inseridos na mesma página e o usário poderá rolar a tela para poder visualizá-los. Em primeiro lugar, é disposto as informações da primeira solução do casamento, com o resultado textual, confome indicado na Figura 2, e com resultado de imagem, na Carta de Smith observado na Figura 4.

Em segundo lugar aparece a outra solução, com suas respectivas informações textuais e Carta de Smith, assim observados nas Figuras 3 e 5.

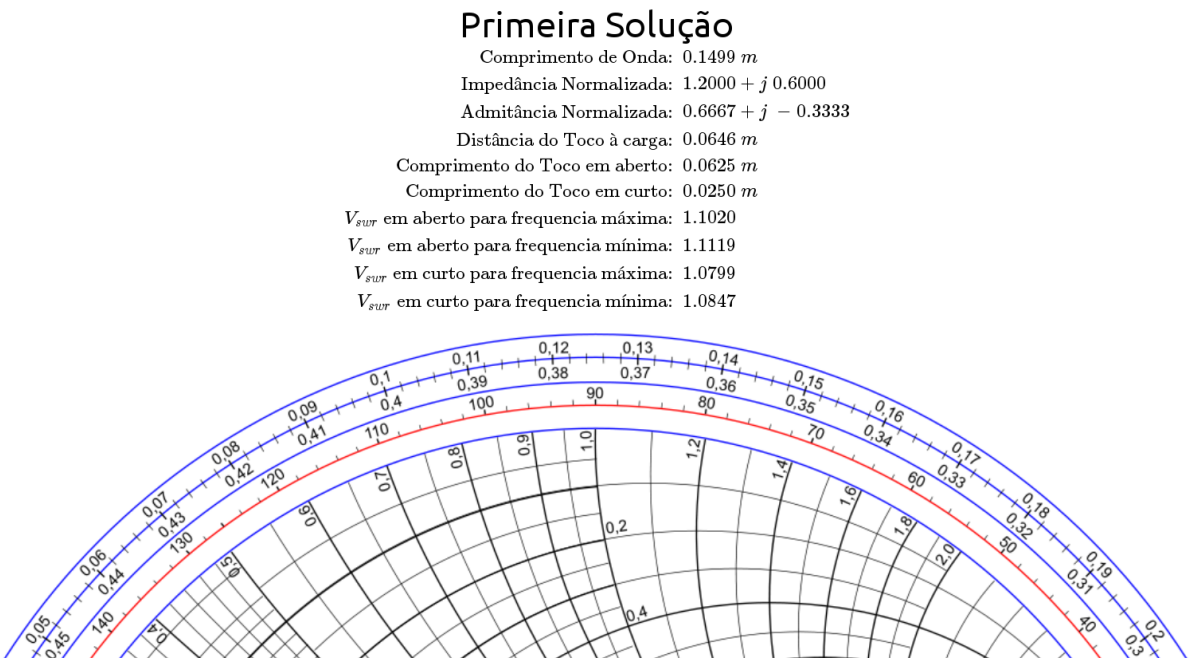

Figura 2 – Indicação textual da primeira solução do casamento.

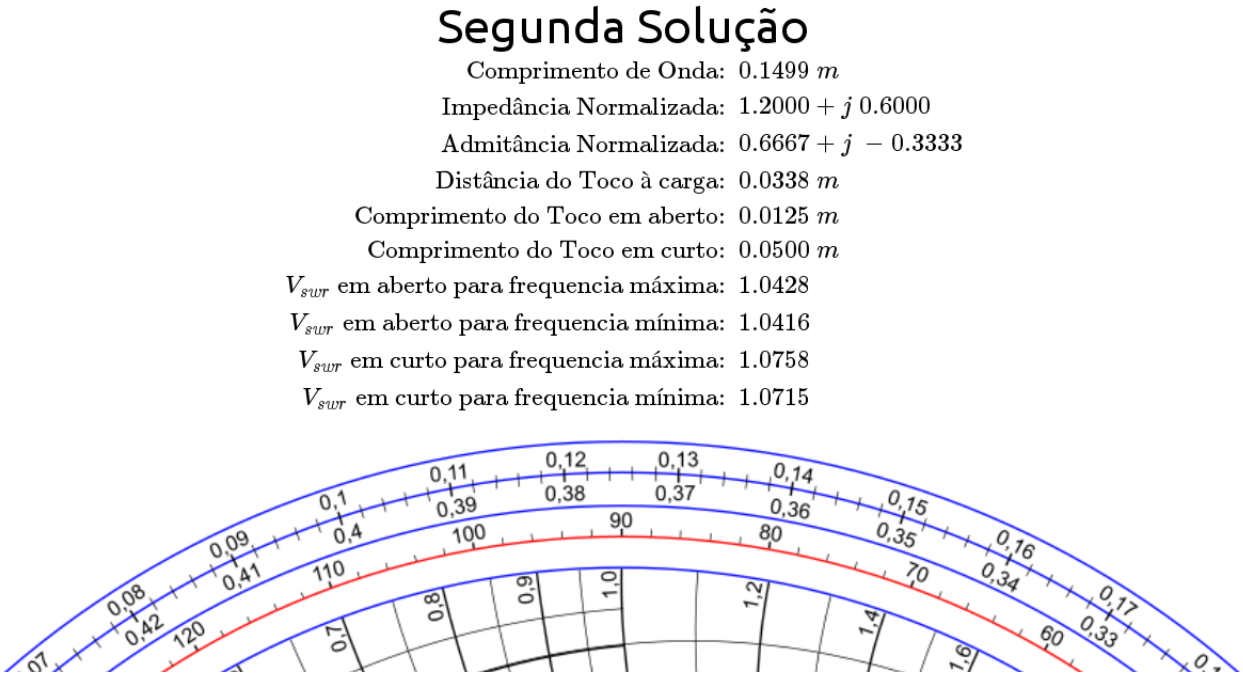

Figura 3 – Indicação textual da segunda solução do casamento.

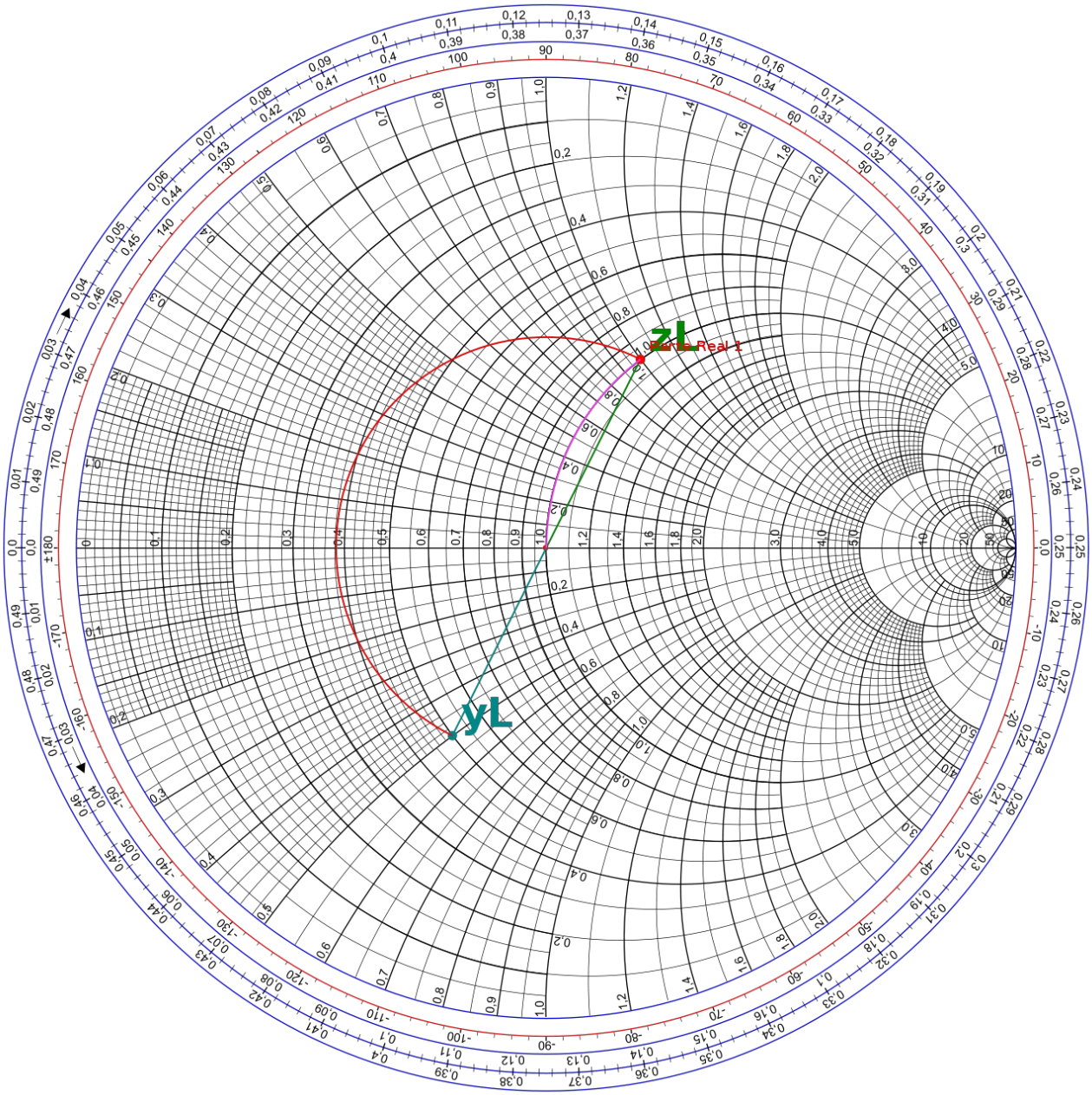

Figura 4 – Carta de Smith da primeira solução do casamento.

Na Carta de Smith, é importante notar os pontos z_L (impedância da carga normalizada) e y_L (admitância da carga normalizada), apresentados em verde e verde marinho, respectivamente. O círculo de ρ constante (módulo do coeficiente de reflexão) também é tracado, em segmento, em vermelho. Esse segmento representa o comprimento elétrico percorrido até que seja atingido o círculo de parte real unitária, cujo ponto também é representado em vermelho.

Dessa foma, com a adição do toco simples, percorre-se a curva em magenta, atingindo o centro da Carta de Smith, o que caracteriza um casamento perfeito. A primeira solução para essa entrada deu-se no segundo cruzamento do círculo de parte real unitária, já a segunda solução, no primeiro cruzamento.

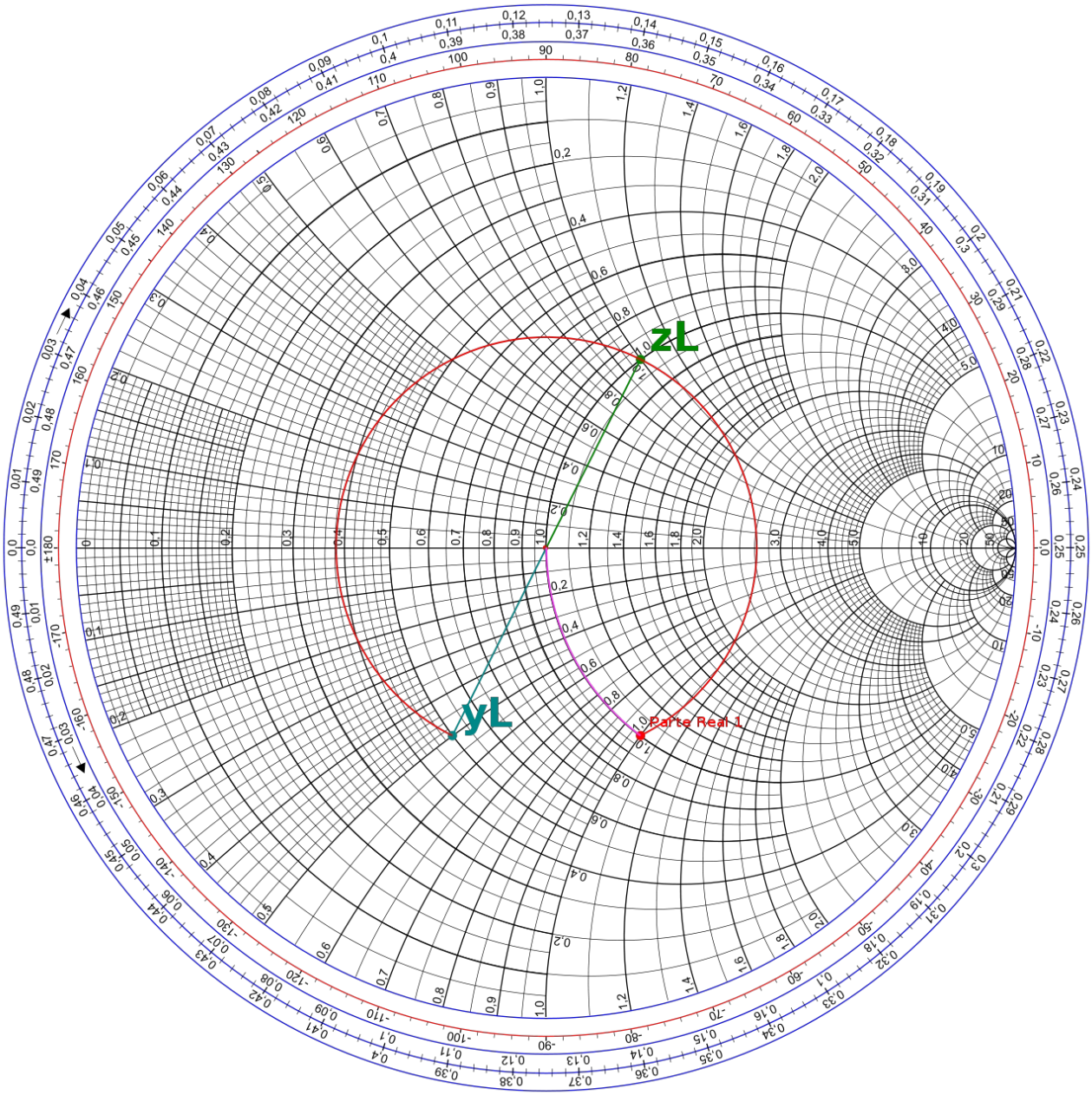

Figura 5 - Carta de Smith da segunda solução do casamento.

O usuário também pode clicar com o botão direito do mouse nas figuras e selecionar a opção “Salvar imagem como” para poder baixá-las na sua resolução máxima de 1300 por 1300 pixels de largura e altura.

Finalmente, tem-se o gráfico com a análise da largura de faixa do casamento, que traça as curvas de SWR em função da frequência de operação. São traçadas 4 curvas referentes aos casamentos com toco em aberto e em curto para a primeira solução e também para o casamento da segunda solução com toco em aberto e em curto. O gráfico gerado para as entradas expostas na Figura 1 pode ser observado na Figura 6. É importante notar que a escala dos valores aparecem representados por B, que indica bilhão. Assim a frequência de 2B Hz é de 2 MHz.

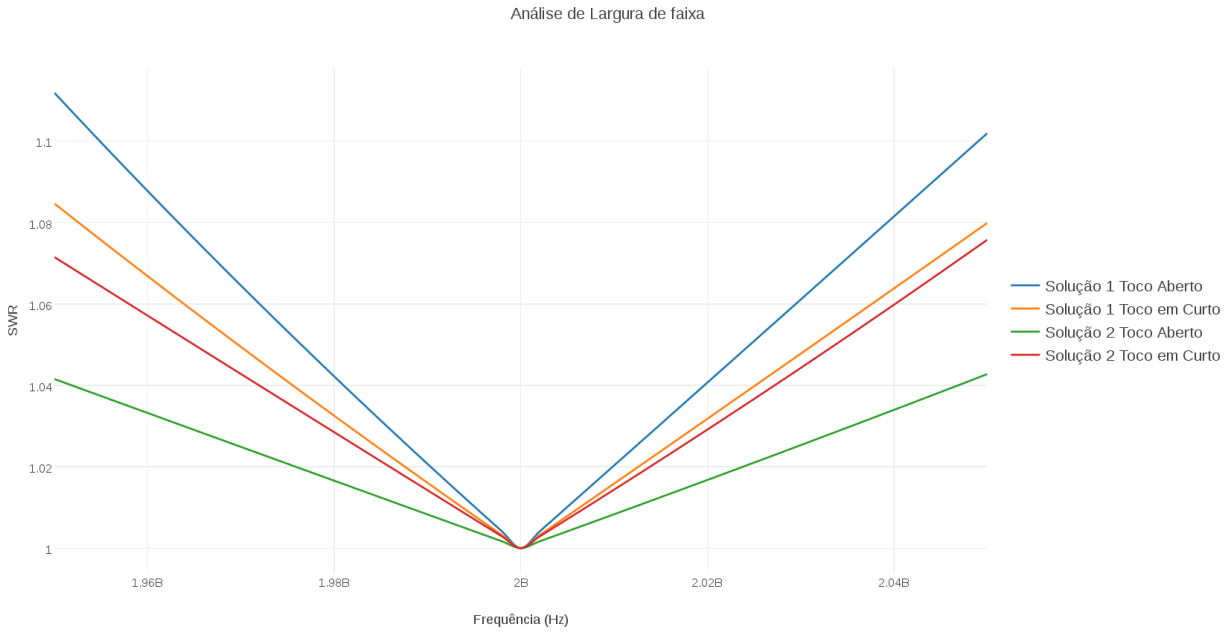

Figura 6 – Gráfico gerado para a análise de largura de faixa.

A Carta de Smith pode ser observada junto com o texto que descreve a sua solução, como indicado na Figura 7, uma vez que o zoom da página pode ser ajustado.

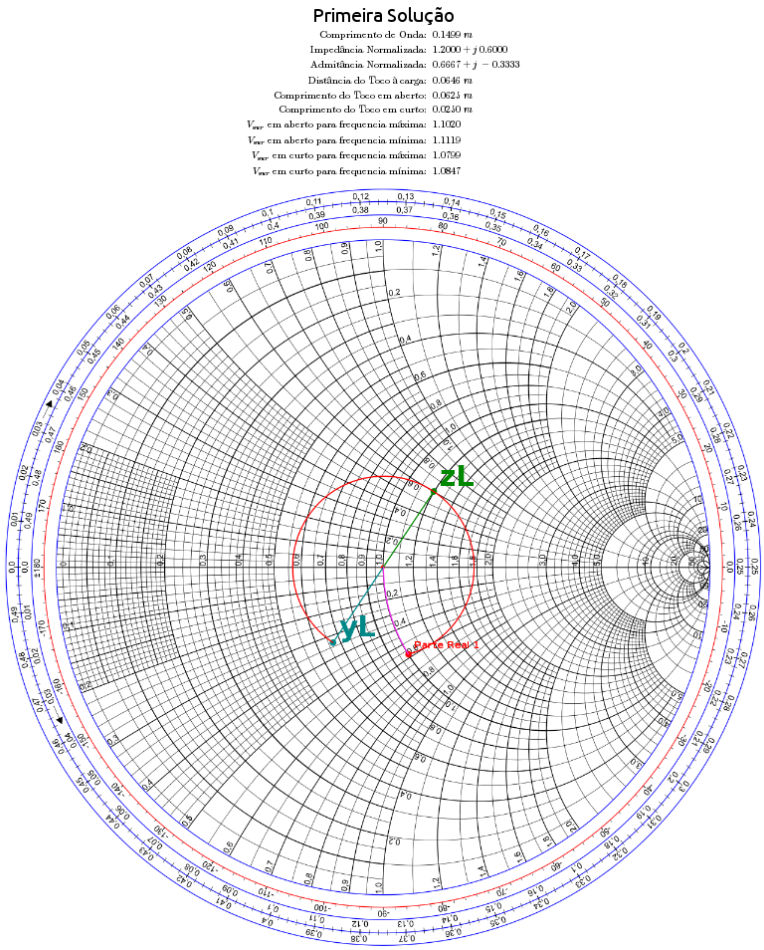

Figura 7 – Conjunto do texto com a Carta de Smith.

Conclusão

O presente trabalho, acompanhado de seus arquivos web, realiza o cálculo do comprimento do toco (em curto ou em aberto) e da distância do toco à carga, indicando as duas soluções. O algoritmo foi testado para diversos casos, e comparados com o material de referência. O resultado final foi um código robusto, de precisão maior que 8 algarismos significativos e que observa os casos nos quais poderiam ocorrer divisão por zero.

Em um primeiro momento, foi observado uma instabilidade computacional quando o denominador de uma divisão se aproximava de zero, o que gerava erros nos cálculos ou resultados que não podiam ser representados numericamente. Depois de extensiva análise e testes, o erro foi sanado e o aplicativo se tornou estável para todas as entradas testadas.

Além de sua robustez, esse aplicativo web possui, portanto, grande potencial didático, uma vez que diferentemente de uma planilha, a sua interface gráfica é amigável, é possível visualizar o desenho da Carta de Smith em alta resolução com os traçados e contornos dos pontos e também há o gráfico com a análise de banda por meio do SWR.

O aplicativo cumpriu com sucesso os seus requisitos de projetos, mas também, como forma de continuidade, ele pode ser aprimorado para poder realizar outros tipos casamentos e ainda permitir que o usuário faça, manualmente, um casamento, com a possibilidade de inserir um número arbitrário de tocos em posições arbitrárias.

Anexo

Código 1:

/home/gabrui/arquivos/ITA/5S/ELE-12/Lab1/lab1.html

```
 1 <!DOCTYPE html>
 2
 3 <html>
 4
 5 <head>
 6     <title>Lab1</title>
 7     <meta charset="UTF-8">
 8     <meta name="viewport" content="width=device-width, initial-scale=1.0">
 9     <script src="math.js" type="text/javascript"></script>
10     <script src="src/plotly-latest.min.js"></script>
11     <script src="codigo.js" type="text/javascript"></script>
12     <script src="cartaSmith.js" type="text/javascript"></script>
13     <script type="text/x-mathjax-config">
14         MathJax.Hub.Config({
15             tex2jax: {inlineMath: [['$','$'], ['\\(','\\)']]}
16         });
17     </script>
18     <script type="text/javascript" async src="path-to-mathjax/MathJax.js?
config=TeX-AMS_CHTML"></script>
19     <script type="text/javascript" async
20     src="https://cdnjs.cloudflare.com/ajax/libs/mathjax/2.7.1/MathJax.js?
config=TeX-MML-AM_CHTML">
21     </script>
22     <link rel="stylesheet" href="estilo.css">
23     <link href="https://fonts.googleapis.com/css?family=Ubuntu"
rel="stylesheet">
24 </head>
25
26
27 <body>
28 <h1> Lab 1 - ELE-12 - Casamento de impedância com toco simples em
paralelo</h1>
29 <p> Especifique as entradas para o cálculo do casamento com toco simples em
paralelo:</p>
30 <form id="entradas" action="">
31   <table>
32       <tr>
33         <td>$Z_0$</td>
34         <td><input type="number" name="ReZ0" step="any" value="50"> <!--+ j
<input type="number" name="ImZ0" step="any" value="0">--></td>
35       </tr>
36       <tr>
37         <td>$Z_L$</td>
38         <td><input type="number" name="ReZL" step="any" value="80"> + $j$
<input type="number" name="ImZL" step="any" value="30"></td>
39       </tr>
40       <tr>
41         <td>$\text{Frequência de Casamento}\;\; (MHz)$</td>
42         <td><input type="number" name="freq" step="any" value="2000"></td>
43       </tr>
44       <tr>
45         <td>$U_{rel}$</td>
46         <td><input type="number" name="urel" step="any" value="1"></td>
47       </tr>
48       <tr>
```

```
        <td>$\epsilon_{rel}$</td>
        <td><input type="number" name="erel" step="any" value="1"></td>
      </tr>
      <tr>
        <td>$\text{BW de operação}\;\; (MHz)$</td>
        <td><input type="number" name="bw" step="any" value="300"></td>
      </tr>
    </table>
</form>

<p>Clique em calcular para obter as respostas.</p>

<button onclick="calcular()">Calcular</button>
<p></p>
<img src="src/drawing.png" alt="Toco Aberto em Paralelo" width="500"/>
<p></p><p></p>
<div id="respostas"></div>

<canvas id="desenho" width="1300" height="1300">
    <img id="fundo" src="src/fundo.svg" alt="Carta de Smith"/>
</canvas>

<div id="respostas2"></div>

<canvas id="desenho2" width="1300" height="1300">
</canvas>

<div id="swr" style="width: 1300px; height: 700px; margin: auto"></div>

</body>
</html>
```

Código 2:

/home/gabrui/arquivos/ITA/5S/ELE-12/Lab1/estilo.css

```
* {
    text-align: center;
}

td:nth-child(1) {
    text-align: right;
}

td:nth-child(2) {
    text-align: left;
}

table {
    margin: auto;
}
h1, p {
font-family: 'Ubuntu', sans-serif;
}

caption {
    font-size: 2em;
    font-family: 'Ubuntu', sans-serif;
}
```

Código 3:

/home/gabrui/arquivos/ITA/5S/ELE-12/Lab1/codigo.js

```
  1 //UTILIZA A BIBLIOTECA math.js para o cálculo com números complexos
  2
  3 /* global math, MathJax, Plotly */
  4
  5 var epsilon = math.pow(10, -8);
  6
  7 /**
  8  * @function somarSeNegativo Soma um incremento a um número se este for
negativo
  9  * @param {Number} incremento Soma esse incremento ao número
 10  * @param {Number} numero Número a ser analisado se negativo
 11  * @returns {Number} Resultado da operação de soma, se numero for negativo
 12  */
 13 function somarSeNegativo(incremento, numero) {
 14     if (numero < 0) {
 15         return numero + incremento;
 16     }
 17     return numero;
 18 }
 19
 20
 21 /**
 22  * @function calcularNovoyToco calcula a admitancia do toco para um valor de
frequencia
 23  * @param {Number} aberto variavel boolena que diz se o toco esta em aberto
ou nao
 24  * @param {Number} l comprimento do tooc
 25  * @param {Number} f valor de frequencia
 26  * @param {Number} u valor da permeabilidade magnetica
 27  * @param {Number} e valor de permissividade eletrica
 28  * @returns {Number} Retorna o valor da admitancia do toco
 29  */
 30 function calcularNovoyToco(aberto, l, f, u,e) {
 31     if (aberto){
 32         return math.tan(2*math.pi*l*f*math.sqrt(u*e));
 33     }
 34     else{
 35         return -1*math.cot(2*math.pi*l*f*math.sqrt(u*e));
 36     }
 37 }
 38
 39
 40 /**
 41  * @function calcularModR Calcula o modulo do coeficiente de reflexao
 42  * @param {Number} a parte real da admitancia de carga
 43  * @param {Number} b parte imaginaria da admitancia de carga
 44  * @param {Number} NyT valor da admitancia do toco
 45  * @param {Number} x posicao do toco
 46  * @param {Number} f frequencia
 47  * @param {Number} u permeabilidade magnetica
 48  * @param {Number} e permissividade eletrica
 49  * @returns {Number}
 50  */
 51 function calcularModR(a,b,NyT,x,f,u,e){
 52     var t = math.tan(2*math.pi*x*f*math.sqrt(u*e));
 53     var t2 = t*t;
 54     var t3 = t2*t;
 55     var t4 = t2*t2;
 56     var Rey = (a - a*b*t + t*a*(b+t))/((1-b*t)*(1-b*t)+(t*a)*(t*a));
 57     var Imy = (-t*a*a+(b+t)*(1-b*t))/((1-b*t)*(1-b*t)+(t*a)*(t*a));
```

```
 58     var ReyLf = (Rey);
 59     var ImyLf = (-NyT+Imy);
 60     var num2 = (1-ReyLf)*(1-ReyLf) + (ImyLf)*(ImyLf);
 61     var dem2 = (1+ReyLf)*(1+ReyLf) + (ImyLf)*(ImyLf);
 62     return math.sqrt(num2/dem2);
 63 }
 64 
 65 
 66 /**
 67  * @function calcularVswr Retorna o VSWR a partir do coeficiente de reflexão
 68  * @param {Number} R Módulo do coeficiente de Reflexão
 69  * @returns {Number} VSWR calculado
 70  */
 71 function calcularVswr(R) {
 72     return (1+R)/(1-R);
 73 }
 74 
 75 
 76 /**
 77  *
 78  * @param {math.complex} zlComplexo A impedância da carga normalizada.
 79  * @returns {math.complex} O coeficiente de reflexão complexo
 80  */
 81 function calcularReflexao(zlComplexo) {
 82     return math.divide(math.subtract(zlComplexo, 1), math.add(1,
zlComplexo));
 83 }
 84 
 85 
 86 
 87 /**
 88  * Gira no sentido horário para o cáculo da diferença de ângulo
 89  * @param {number} inicio Ângulo inicial entre -pi e pi
 90  * @param {number} fim Ângulo final entre -pi e pi
 91  * @returns {number} Distância angular a ser percorrida no sentido horário
 92  */
 93 function calcDifAngRad(inicio, fim) {
 94     // Tem que passar pela descontinuidade
 95     if (inicio < fim) {
 96         return 2*math.pi - (fim - inicio);
 97     } else {
 98         return inicio - fim;
 99     }
100 }
101 
102 
103 
104 function calcular() {
105 
106     var ReZ0 = document.getElementsByName("ReZ0")[0].value;
107     var ImZ0 = 0;//document.getElementsByName("ImZ0")[0].value;
108     var Z0 = math.complex(ReZ0, ImZ0);
109     var ReZL = document.getElementsByName("ReZL")[0].value;
110     var ImZL = document.getElementsByName("ImZL")[0].value;
111     var ZL = math.complex(ReZL, ImZL);
112     var freq = document.getElementsByName("freq")[0].value * math.pow(10,
6);
113     var urel = document.getElementsByName("urel")[0].value;
114     var erel = document.getElementsByName("erel")[0].value;
115     var bw   = document.getElementsByName("bw")  [0].value * math.pow(10,6);
116 
117     var u = urel * 4 * math.pi * math.pow(10, -7);
118     var e = erel * 8.8541878176 * math.pow(10, -12);
119     // Impedância Normalizada da carga
```

```
120     var ZLNorm = math.divide(ZL, Z0);
121     // Admitância Normalizada da carga
122     var YLNorm = math.divide(1, ZLNorm);
123     // Comprimento de onda
124     var lambda = 1/(math.sqrt(u*e)*freq);
125     // Coeficiente de reflexão do toco em curto normalizada
126     var rTocoCurto = math.complex(-1, 0);
127     // Admitância do toco em aberto normalizada
128     var rTocoAberto = math.complex(1, 0);
129     // Frequência máxima da banda de operação
130     var fmax = freq + bw/2;
131     // Frequência mínima da banda de operação
132     var fmin = freq - bw/2;
133     var freqs = [fmin, fmax];
134
135
136     /* CÓDIGO PARA CALCULAR PELA CARTA DE SMITH
137     for (var numSol = 1; numSol <=2; numSol++) {
138         // Coeficiente de Reflexão da carga, complexo
139         var reflexaoCarga = calcularReflexao(ZLNorm);
140         // O seu módulo
141         var reflexaoCargaModulo = reflexaoCarga.toPolar().r;
142         // O coeficiente de reflexão no ponto de parte real da admitância
igual
143         // a 1. Ele é simétrico com relação a outra solução
144         var reflexaoParteReal1 = math.complex.fromPolar(reflexaoCargaModulo,
145             math.acos(reflexaoCargaModulo) * ((numSol===1)?1:-1));
146         // A admitância desse ponto de reflexão
147         var admitanciaParteReal1 = math.divide(math.add(1,
reflexaoParteReal1),
148             math.subtract(1, reflexaoParteReal1));
149         // Um coeficiente de reflexão associada à admitância da carta de
smith
150         var reflexaoAdmitancia = calcularReflexao(YLNorm);
151         // A distância elétrica
152         var xEletrico = calcDifAngRad(reflexaoAdmitancia.toPolar().phi,
153                         reflexaoParteReal1.toPolar().phi)*0.5/(2*math.pi);
154         // A distância real
155         var x = xEletrico * lambda;
156         // Comprimento do Toco em curto
157         var compTocoCurto = calcDifAngRad(rTocoCurto.toPolar().phi,
158             -reflexaoParteReal1.toPolar().phi)*lambda*0.5/(2*math.pi);
159         // Comprimento do Toco em aberto
160         var compTocoAberto = calcDifAngRad(rTocoAberto.toPolar().phi,
161             -reflexaoParteReal1.toPolar().phi)*lambda*0.5/(2*math.pi);
162
163         // Descasamento de
164         for (var tam = freqs.length - 1; tam >= 0; tam--) {
165
166         }
167     }*/
168
169     var a = YLNorm.re;
170     var b = YLNorm.im;
171
172
173     var A = a*a + b*b - a;
174     var B = -2*b;
175     var C = 1 - a;
176     var delta = B*B - 4*A*C;
177     if (math.abs(A) < epsilon) {
178         var t2 = -C/B;
179
180         var x1 = lambda/4;
```

```
181         var x2 = somarSeNegativo(lambda/2, lambda * math.atan(t2)/
(2*math.pi));
182
183         var Imy1 =  ZLNorm.im;
184         var Imy2 = -ZLNorm.im;
185     } else {
186         var t1 = (-B + math.sqrt(delta))/(2*A);
187         var t2 = (-B - math.sqrt(delta))/(2*A);
188
189         var x1 = somarSeNegativo(lambda/2, lambda * math.atan(t1)/
(2*math.pi));
190         var x2 = somarSeNegativo(lambda/2, lambda * math.atan(t2)/
(2*math.pi));
191
192         var Imy1 = ((b+t1)*(1-b*t1)-t1*a*a)/((1-b*t1)*(1-b*t1)+
(t1*a)*(t1*a));
193         var Imy2 = ((b+t2)*(1-b*t2)-t2*a*a)/((1-b*t2)*(1-b*t2)+
(t2*a)*(t2*a));
194     }
195
196
197     var comp1aberto = somarSeNegativo(lambda/2, math.atan(Imy1)*lambda/
(2*math.pi));
198     var comp1curto = somarSeNegativo(lambda/2, math.atan(-1/Imy1)*lambda/
(2*math.pi));
199     var comp2aberto = somarSeNegativo(lambda/2, math.atan(Imy2)*lambda/
(2*math.pi));
200     var comp2curto = somarSeNegativo(lambda/2, math.atan(-1/Imy2)*lambda/
(2*math.pi));
201
202
203     var f = fmin;
204     var quantIter = 50;
205     var df = bw/quantIter;
206     var xGrafico = [];
207     var Vswr1aberto = [];
208     var Vswr2aberto = [];
209     var Vswr1curto = [];
210     var Vswr2curto = [];
211     for (var quant = 0; quant <= quantIter; quant++, f+=df) {
212         var NyTaberto1 = calcularNovoyToco(true,comp1aberto,f,u,e);
213         var NyTaberto2 = calcularNovoyToco(true,comp2aberto,f,u,e);
214         var NyTcurto1 = calcularNovoyToco(false,comp1curto,f,u,e);
215         var NyTcurto2 = calcularNovoyToco(false,comp2curto,f,u,e);
216
217         var R1aberto = calcularModR(a,b,NyTaberto1,x1,f,u,e);
218         var R2aberto = calcularModR(a,b,NyTaberto2,x2,f,u,e);
219
220         var R1curto = calcularModR(a,b,NyTcurto1,x1,f,u,e);
221         var R2curto = calcularModR(a,b,NyTcurto2,x2,f,u,e);
222
223         xGrafico[quant] = f;
224         Vswr1aberto[quant] = calcularVswr(R1aberto);
225         Vswr2aberto[quant] = calcularVswr(R2aberto);
226         Vswr1curto[quant] = calcularVswr(R1curto);
227         Vswr2curto[quant] = calcularVswr(R2curto);
228     }
229
230     var texto = "";
231     var carta1 = new CartaSmith(649, 649, 1196-648, "desenho", "fundo");
232
233     texto += "<table>";
234     texto += "<caption>Primeira Solução</caption>";
235     texto += "<tr><td>$\\text{Comprimento de Onda: }$</td><td>$" +
```

```
lambda.toFixed(4) + "\\;m $</td></tr>";
236     texto += "<tr><td>$\\text{Impedância Normalizada: }$</td><td>$" +
ZLNorm.re.toFixed(4) +" + j\\;" + ZLNorm.im.toFixed(4) + "$</td></tr>";
237     carta1.desenharRetaZNorm(ZLNorm);
238     carta1.desenharPontoZNorm(ZLNorm, "zL");
239     texto += "<tr><td>$\\text{Admitância Normalizada: }$</td><td>$" +
YLNorm.re.toFixed(4) +" + j\\;" + YLNorm.im.toFixed(4) + "$</td></tr>";
240     carta1.setCor("#008888");
241     carta1.desenharRetaZNorm(YLNorm);
242     carta1.desenharPontoZNorm(YLNorm, "yL");
243     carta1.setCor("#FF0000");
244     carta1.curvaRConst(calcularReflexao(YLNorm),
calcularReflexao(math.complex(1, Imy1)));
245     carta1.setFonte("1em");
246     carta1.desenharPontoZNorm(math.complex(1, Imy1), "Parte Real 1");
247     carta1.setCor("#FF33FF");
248     carta1.interpolarZ(math.complex(1, Imy1), math.complex(1, 0));
249     texto += "<tr><td>$\\text{Distância do Toco à carga: }$</td><td>$" +
x1.toFixed(4) +"\\;m $</td></tr>";
250     texto += "<tr><td>$\\text{Comprimento do Toco em aberto: }$</td><td>$" +
comp1aberto.toFixed(4) +"\\;m $</td></tr>";
251     texto += "<tr><td>$\\text{Comprimento do Toco em curto: }$</td><td>$" +
comp1curto.toFixed(4) +"\\;m $</td></tr>";
252     texto +="<tr><td>$V_{swr} \\text{ em aberto para frequencia máxima: }
$</td><td>$"+Vswr1aberto[quantIter].toFixed(4)+"\\;\\; $</td></tr>";
253     texto +="<tr><td>$V_{swr} \\text{ em aberto para frequencia mínima: }
$</td><td>$"+Vswr1aberto[0].toFixed(4)+"\\;\\; $</td></tr>";
254     texto +="<tr><td>$V_{swr} \\text{ em curto para frequencia máxima: }
$</td><td>$"+Vswr1curto[quantIter].toFixed(4)+"\\;\\; $</td></tr>";
255     texto +="<tr><td>$V_{swr} \\text{ em curto para frequencia mínima: }
$</td><td>$"+Vswr1curto[0].toFixed(4)+"\\;\\; $</td></tr>";
256     texto += "</table>";
257
258
259     document.getElementById("respostas").innerHTML = texto;
260     MathJax.Hub.Queue(["Typeset",MathJax.Hub,"respostas"]);
261
262
263     var texto2 = "";
264     var carta2 = new CartaSmith(649, 649, 1196-648, "desenho2", "fundo");
265     texto2 += "<table>";
266     texto2 += "<caption>Segunda Solução</caption>";
267     texto2 += "<tr><td>$\\text{Comprimento de Onda: }$</td><td>$" +
lambda.toFixed(4) + "\\;m $</td></tr>";
268     texto2 += "<tr><td>$\\text{Impedância Normalizada: }$</td><td>$" +
ZLNorm.re.toFixed(4) +" + j\\;" + ZLNorm.im.toFixed(4) + "$</td></tr>";
269     carta2.desenharRetaZNorm(ZLNorm);
270     carta2.desenharPontoZNorm(ZLNorm, "zL");
271     texto2 += "<tr><td>$\\text{Admitância Normalizada: }$</td><td>$" +
YLNorm.re.toFixed(4) +" + j\\;" + YLNorm.im.toFixed(4) + "$</td></tr>";
272     carta2.setCor("#008888");
273     carta2.desenharRetaZNorm(YLNorm);
274     carta2.desenharPontoZNorm(YLNorm, "yL");
275     carta2.setCor("#FF0000");
276     carta2.curvaRConst(calcularReflexao(YLNorm),
calcularReflexao(math.complex(1, Imy2)));
277     carta2.setFonte("1em");
278     carta2.desenharPontoZNorm(math.complex(1, Imy2), "Parte Real 1");
279     carta2.setCor("#FF33FF");
280     carta2.interpolarZ(math.complex(1, Imy2), math.complex(1, 0));
281     texto2 += "<tr><td>$\\text{Distância do Toco à carga: }$</td><td>$" +
x2.toFixed(4) +"\\;m $</td></tr>";
282     texto2 += "<tr><td>$\\text{Comprimento do Toco em aberto: }$</td><td>$"
+ comp2aberto.toFixed(4) +"\\;m $</td></tr>";
```

```
283     texto2 += "<tr><td>$\\text{Comprimento do Toco em curto: }$</td><td>$" +
comp2curto.toFixed(4) +"\\;m $</td></tr>";
284     texto2 +="<tr><td>$V_{swr} \\text{ em aberto para frequencia máxima: }
$</td><td>$"+Vswr2aberto[quantIter].toFixed(4)+"\\;\\; $</td></tr>";
285     texto2 +="<tr><td>$V_{swr} \\text{ em aberto para frequencia mínima: }
$</td><td>$"+Vswr2aberto[0].toFixed(4)+"\\;\\; $</td></tr>";
286     texto2 +="<tr><td>$V_{swr} \\text{ em curto para frequencia máxima: }
$</td><td>$"+Vswr2curto[quantIter].toFixed(4)+"\\;\\; $</td></tr>";
287     texto2 +="<tr><td>$V_{swr} \\text{ em curto para frequencia mínima: }
$</td><td>$"+Vswr2curto[0].toFixed(4)+"\\;\\; $</td></tr>";
288     texto2 += "</table>";
289
290
291     document.getElementById("respostas2").innerHTML = texto2;
292     MathJax.Hub.Queue(["Typeset",MathJax.Hub,"respostas2"]);
293
294
295     var dadosVswr1a = {
296         x: xGrafico,
297         y: Vswr1aberto,
298         mode: 'lines',
299         name: 'Solução 1 Toco Aberto',
300         line: {shape: 'spline'},
301         type: 'scatter'
302     };
303     var dadosVswr2a = {
304         x: xGrafico,
305         y: Vswr2aberto,
306         mode: 'lines',
307         name: 'Solução 2 Toco Aberto',
308         line: {shape: 'spline'},
309         type: 'scatter'
310     };
311     var dadosVswr1c = {
312         x: xGrafico,
313         y: Vswr1curto,
314         mode: 'lines',
315         name: 'Solução 1 Toco em Curto',
316         line: {shape: 'spline'},
317         type: 'scatter'
318     };
319     var dadosVswr2c = {
320         x: xGrafico,
321         y: Vswr2curto,
322         mode: 'lines',
323         name: 'Solução 2 Toco em Curto',
324         line: {shape: 'spline'},
325         type: 'scatter'
326     };
327
328     var dados = [dadosVswr1a, dadosVswr1c, dadosVswr2a, dadosVswr2c];
329
330     var estilo = {
331         legend: {
332             y: 0.5,
333             font: {size: 16},
334             yref: 'paper'
335         },
336         title: 'Análise de Largura de faixa',
337         xaxis: {
338             title: 'Frequência (Hz)'
339         },
340         yaxis: {
341             title: 'SWR'
```

```
342         }
343     };
344 
345     Plotly.newPlot('swr', dados, estilo);
346 
347 
348 }
349 
350 
```

Código 4:

/home/gabrui/arquivos/ITA/5S/ELE-12/Lab1/cartaSmith.js

```
  1 /* global math */
  2
  3 //UTILIZA FUNÇÕES DO ARQUIVO codigo.js
  4
  5 /**
  6  * @class Representação gráfica da Carta de Smith
  7  * @param {Number} cx A coordenada x do centro da carta de smith da imagem
  8  * @param {Number} cy A coordenada y do centro da carta de smith da imagem
  9  * @param {Number} r O raio da carta de smith da imagem
 10  * @param {string} desenho O ID do canvas no html
 11  * @param {string} fundo O ID da imagem de fundo do html
 12  * @returns {CartaSmith} Um objeto do tipo carta de smith
 13  */
 14 CartaSmith = function (cx, cy, r, desenho, fundo) {
 15
 16     /**
 17      *
 18      * @param {Number} cx A coordenada x do centro da carta de smith da
imagem
 19      * @param {Number} cy A coordenada y do centro da carta de smith da
imagem
 20      * @param {Number} r O raio da carta de smith da imagem
 21      * @param {string} desenho O ID do canvas no html
 22      * @param {string} fundo O ID da imagem de fundo do html
 23      */
 24     this.iniciar = function(cx, cy, r, desenho, fundo) {
 25         this.cx = cx; ///< Posição x do centro da figura
 26         this.cy = cy; ///< Posição y do centro da figura
 27         this.r = r;   ///< Raio da carta de smith da figura
 28         this.canvas = document.getElementById(desenho);
 29         this.desenho = this.canvas.getContext("2d");
 30         this.fundo = document.getElementById(fundo);
 31         this.desenho.restore();
 32         this.apagar();
 33         this.desenho.moveTo(0, 0);
 34         this.setCor("#008800");
 35         this.desenho.lineWidth = 2;
 36         this.fonte = "'Open Sans', sans-serif";
 37         this.desenho.font = "bold 3em 'Open Sans', sans-serif";
 38     };
 39
 40     /**
 41      * Muda a cor de desenho dos pontos e retas da carta
 42      * @param {string} cor A representação hexadecimal ou de nome da cor
 43      */
 44     this.setCor = function(cor) {
 45         this.desenho.fillStyle = cor;
 46         this.desenho.strokeStyle = cor;
 47     };
 48
 49
 50     /**
 51      * Muda a fonte
 52      * @param {type} fonte A representação da fonte
 53      */
 54     this.setFonte = function(fonte) {
 55         this.desenho.font = "bold " + fonte + this.fonte;
 56     };
 57
 58
```

```
 59     /**
 60      * @function apagar Apaga que foi desenhado
 61      */
 62     this.apagar = function() {
 63         this.desenho.clearRect(0, 0, this.canvas.width, this.canvas.height);
 64         this.desenho.drawImage(this.fundo, 0, 0);
 65     };
 66
 67
 68     /**
 69      * @function desenharPontoZNorm Desenha um ponto de impedância
normalizada
 70      * na carta de smith.
 71      * @param {math.complex} zNormComplexo Impedância normalizada
 72      * @param {string} texto Comentário sobre o ponto
 73      */
 74     this.desenharPontoZNorm = function(zNormComplexo, texto) {
 75         this.desenharR(calcularReflexao(zNormComplexo), texto);
 76     };
 77
 78
 79     /**
 80      * @function desenharRetaR Desenha uma reta que liga o centro da carta
de
 81      * smith até o ponto que representa uma impedância normalizada.
 82      * @param {math.complex} zNormComplexo Impedância normalizada
 83      */
 84     this.desenharRetaZNorm = function(zNormComplexo) {
 85         this.desenharRetaR(calcularReflexao(zNormComplexo));
 86     };
 87
 88
 89     /**
 90      * @function desenharR Desenha um ponto na carta de Smith referente a o
 91      * coeficiente de reflexão dado
 92      * @param {math.complex} reflexaoComplexo Coeficiente de reflexão
 93      * @param {string} texto Texto a ser legenda do ponto
 94      */
 95     this.desenharR = function(reflexaoComplexo, texto) {
 96         var x =  reflexaoComplexo.re*this.r + this.cx; // Posição x na
figura
 97         var y = -reflexaoComplexo.im*this.r + this.cy; // Posição y na
figura
 98         this.desenho.beginPath();
 99         this.desenho.arc(x, y, this.r/100, 0, 2*Math.PI);
100         this.desenho.fill();
101         if (texto) {
102             this.desenho.fillText(texto, x + 10, y - 10);
103         }
104     };
105
106
107     /**
108      * @function desenharRetaR Desenha uma reta que liga o centro da carta
de
109      * smith até o ponto de coeficiente de reflexão especificado.
110      * @param {math.complex} reflexaoComplexo Coeficiente de reflexão
111      */
112     this.desenharRetaR = function(reflexaoComplexo) {
113         var x =  reflexaoComplexo.re*this.r + this.cx; // Posição x na
figura
114         var y = -reflexaoComplexo.im*this.r + this.cy; // Posição y na
figura
115         this.desenho.beginPath();
```

```
116         this.desenho.moveTo(this.cx, this.cy);
117         this.desenho.lineTo(x, y);
118         this.desenho.stroke();
119     };
120 
121 
122     /**
123      *
124      * @param {math.complex} zNormInicial Z Normalizado inicial
125      * @param {math.complex} zNormFinal Z Normalizado final
126      */
127     this.interpolarZ = function(zNormInicial, zNormFinal) {
128         var i = 0;
129         var iteracoes = 100;
130         var delta = math.divide(math.subtract(zNormFinal, zNormInicial),
iteracoes);
131         var temp = zNormInicial.clone();
132         var rTemp = calcularReflexao(temp);
133         var x =  rTemp.re*this.r + this.cx; // Posição x na figura
134         var y = -rTemp.im*this.r + this.cy; // Posição y na figura
135         this.desenho.beginPath();
136         this.desenho.moveTo(x, y);
137         for(i = 0; i<iteracoes; i++) {
138             temp = math.add(temp, delta);
139             rTemp = calcularReflexao(temp);
140             x =  rTemp.re*this.r + this.cx; // Posição x na figura
141             y = -rTemp.im*this.r + this.cy; // Posição y na figura
142             this.desenho.lineTo(x, y);
143         }
144         this.desenho.stroke();
145     };
146 
147 
148     /**
149      * Desenha uma curva, segmento de circunferência de Coeficiente de
Reflexão
150      * constante.
151      * @param {math.complex} rInicial Marca o ponto de início, é o seu raio
que
152      * é utilizado
153      * @param {math.complex} rFinal Marca o ponto de final
154      */
155     this.curvaRConst = function(rInicial, rFinal) {
156         this.desenho.beginPath();
157         this.desenho.arc(this.cx, this.cy, rInicial.toPolar().r * this.r,
158             -rInicial.toPolar().phi, -rFinal.toPolar().phi);
159         this.desenho.stroke();
160     };
161 
162     this.iniciar(cx, cy, r, desenho, fundo);
163 };
```