Tee Matching Program "Teedesign.m"

`Teedesign` is a source code for the `Octave` program, described elsewhere on my website[1].

- The distribution is a zip file containing 5 `Octave` source files (with suffixes "`.m`") and this help file, `Teedesign Explanation.pdf`. Unzip them all into a new folder somewhere. All source files are text files which can be edited by you if you have `Octave` competency.
- Fire up `Octave` and navigate to this folder.
- At the prompt, enter `teedesign`. This runs the program.

Operation:

The program can be run in two modes:

- "Simple" mode, the default, or
- "Expanded" mode, selected by entering "1" at the fourth prompt. This shows more information about processing, and some intermediate results. This mode was mainly used for diagnostic purposes during code development.

Running in "Simple" Mode:

This is probably all you'll need most of the time. The program prompts you for four inputs. If "`<return>` " is given, the default value, shown in brackets, is used.

The program assumes that you wish to see 50 Ω looking into the network. It requests

- The load resistance, Ohm.
- The load reactance, Ohm, positive for inductive, negative for capacitive.
- The design frequency in MHz.
- The inductor Q.
- An "expanded results" query. Entering "`<return>` " selects simple output. Entering "1" (or any number, actually) gives expanded results.

Unlike other matching programs I know, this one starts where you start with a *real* transmatch, by selecting some value of inductance from the front panel. Given the parameters entered, the program estimates the *minimum* value of inductance that you can use. You can then select either this value ("`<return>` ") or a larger value. We see why you might need to do this later. It then finds the capacitance values required for an exact match, the voltages developed across them, and the power dissipated in the (lossy) inductor.

Doing this requires some fancy footwork, expedited by the excellent selection of algorithms provided by `Octave` for matrix inversion, complex number manipulation and function minimisation.

[1] My website is `http://www.qsl.net/zl1an/index.html`. Look for the link under "*Analysing a Tee matching circuit using Octave*" on the *Software* page.

A Sample Dialogue

Assume that we wish to match an inductive load impedance of $z_L = 500 + j500\,\Omega$ at the default frequency, using the default inductor Q, the estimated minimum value of inductance, and "'simple" coutput. Here's what the screen shows:

```
 Load resistance?  (50)  500
 Load reactance?   (zero)  500
 Design frequency, MHz?  (3.5)
 Inductor Q?  (100)
 Expanded results?  (1 for yes, return for no)

 Minimum inductance is  7.26 uH
 Inductance, uH?  (Minimum value)
 Adjusted Zin    =  50.01 +   0.01j
 Load voltage    = 219.15 Volt rms
 Power in load   =  96.06 Watt  (100 Watt in)
 True Power loss =   3.94 Watt

 For matching at 3.50 MHz,
    C1 = 267.43 pF  (Lossless 272.64 pf)
    C2 =  58.49 pF  (Lossless  62.24 pf)
     L =   7.26 uH

 Peak Vc1        = 340.04 Volt
 Peak Vc2        = 171.99 Volt
 Peak Vinductor  = 354.45 Volt
>>
```

Having accepted the inductance value, the program adjusts the capacitors until the input impedance is (very close to) $50\,\Omega$. This value is shown. It will usually have a very small reactive component, which is negligible. It then shows the voltage across, and the power transmitted to the load, and the "true" power loss - which will occur in, and heat the inductor.

Values of the required capacitors are then shown, and in brackets, what they would have been if the inductor was lossless. You'll see that the lossy inductor has necessitated slightly different values.

The peak voltages developed across each capacitor is then shown. Typically, the plate spacing of receiving variable capacitors is about 0.2 - 0.3 mm, which arc over at about 300 - 500 volt rms. Dust, pitting, high spots and high humidity can reduce this. This can be useful to know if high, or highly reactive loads are specified. In this case, we see that the voltage across C_1 may possibly cause arcing with a "receiver-spaced" capacitor.

Of course, you *do* need to know what the impedance of your load is, which is the impedance seen at the input of your coaxial cable or balun. Other modelling programs are available on the web to help you estimate this. Or maybe you can measure it with an antenna analyser. Or maybe you can just make a stab at it.

To run the program again, you'll have to re-enter all the parameters. One day, I may modify the code to change the defaults to those entered last time you ran it, and/or put it in a nice *Windows* form with boxes and buttons. But this will have to wait until `Octave` makes this easier. In the meantime, you're stuck with the `Dos`-like command window.

When the "Minimum inductance" value Fails:

Here is the same dialogue, but the load is now $z_L = 500 - j500\,\Omega$, or capacitive.

```
 Load resistance?  (50)  500
 Load reactance?   (zero)  -500
 Design frequency, MHz?  (3.5)
 Inductor Q?  (100)
 Expanded results?  (1 for yes, return for no)

 Minimum inductance is  7.26 uH
 Inductance, uH?  (Minimum value)
 Adjusted Zin    =  50.01 +   0.01j
 Load voltage    = 219.15 Volt rms
 Power in load   =  96.06 Watt  (100 Watt in)
 True Power loss =   3.94 Watt

 For matching at 3.50 MHz,
    C1 = 267.43 pF  (Lossless 272.64 pf)
    C2 = -204.33 pF  (Lossless -168.79 pf)
     L =   7.26 uH

 Warning! C2 negative!  Increase inductance value

  Peak Vc1        = 340.04 Volt
  Peak Vc2        = 171.99 Volt
  Peak Vinductor  = 354.45 Volt
>>
```

The warning, 4 lines from the dialogue end, alerts us to the fact that the output capacitance value found is negative - or should be an *inductance*. We don't want this, so try again, increasing the inductance to $10\,\mu$H. The last section of the dialogue now reads:

```
 Minimum inductance is  7.26 uH
 Inductance, uH?  (Minimum value)   10
 Adjusted Zin    =  49.99 +  -0.01j
 Load voltage    = 215.98 Volt rms
 Power in load   =  93.30 Watt  (100 Watt in)
 True Power loss =   6.70 Watt

 For matching at 3.50 MHz,
    C1 = 170.24 pF  (Lossless 169.52 pf)
    C2 = 192.69 pF  (Lossless 223.73 pf)
     L =  10.00 uH

  Peak Vc1        = 534.30 Volt
  Peak Vc2        = 449.61 Volt
  Peak Vinductor  = 543.55 Volt
>>
```

This design is acceptable because we end up with two capacitors. The "estimated minimum inductance" value we initially found was unacceptable to us because, although the resulting transmatch design is correct, we cannot realize it without replacing the output capacitor by an inductor. But note

that the voltages across the capacitors are rather high. Try this again using $L = 9\,\mu$H. You'll find that this works too, with lower capacitor voltages.

Selecting the "Expanded" output:

This results in the dialogue and output screen shown below.

```
 Load resistance?  (50)  500
 Load reactance?   (zero)  -500
 Design frequency, MHz?  (3.5)
 Inductor Q?  (100)
 Expanded results?  (1 for yes, return for no)  1

 Minimum inductance is  7.26 uH
 Inductance, uH?  (Minimum value)   10
 Unadjusted Zin       =  53.15 +  -1.22j
 Estimated power loss =   6.77  Watt

 Lossless design values Xc1 = -268.25,   Xc2 = -203.25   XL = 219.91
 Adjusted lossy values  Xc1 = -267.12,   Xc2 = -235.99
 Adjusted Zin    =  49.99 +  -0.01j
 Load voltage    = 215.98 Volt rms
 Power in load   =  93.30 Watt  (100 Watt in)
 True Power loss =   6.70 Watt

 For matching at 3.50 MHz,
    C1 = 170.24 pF  (Lossless 169.52 pf)
    C2 = 192.69 pF  (Lossless 223.73 pf)
     L =  10.00 uH

  Vc1             = 377.81 Volt rms
  Vc2             = 317.92 Volt rms
  Vinductor       = 384.35 Volt rms

  Peak Vc1        = 534.30 Volt
  Peak Vc2        = 449.61 Volt
  Peak Vinductor  = 543.55 Volt

 76 calls to fitting function were required
>>
```

We now see more information. Estimated and corrected values of reactance are shown.The "unadjusted z_{in}" and "estimated power loss" are those computed before the initial estimates of capacitor values have been corrected to the "Adjusted lossy values" shown later. The "true" power loss is usually less than the "estimated power loss". But it's not much different, and the "estimated" loss, which is simple to calculate analytically, see later, is good enough for most purposes.

The "adjusted z_{in}" is the value the optimization algorithm accepted after repeated iteration of the capacitor values. To see a plot of the convergence process, edit the source file and set the parameter "`plotflag`", near the start of the code, to 1.

Some General Rules:

- For minimum power loss for a given inductor Q, select the *smallest* value of inductance that gives realizable capacitor values.
- As the inductor value is increased, the power loss also increases, but the required capacitance values *decrease*. Thus design is a trade-off between these two parameters.

Algorithm Explanation:

High and Low-pass Tee sections can be designed by concatenating two L sections, as shown in the diagram below. Design starts by selecting a value for the "characteristic impedance", R_o, using only the *resistive* part of the load, R_2. The design procedure for matching resistances R_1 and R_2 is:

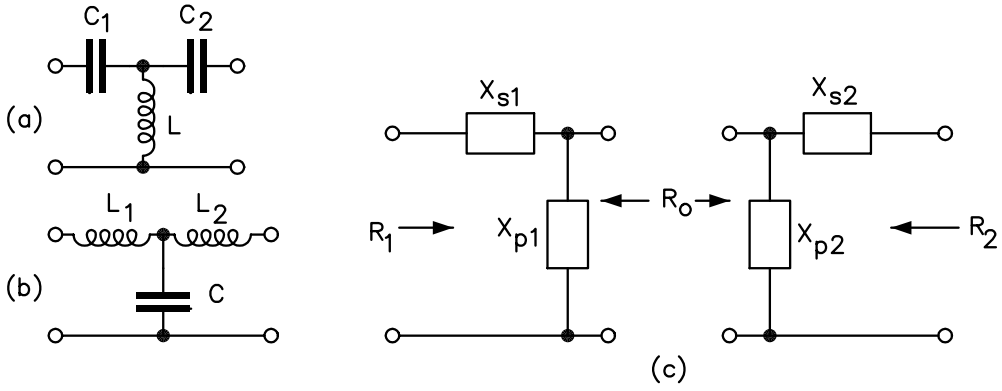

Figure 1: Designing a Tee match using two L sections.

1. Choose R_0 so that $R_0 > R_1$ and $R_0 > R_2$.
2. Calculate $X_{s1} = \pm\sqrt{R_1(R_0 - R_1)}$ and $X_{s2} = \pm\sqrt{R_2(R_0 - R_2)}$, using the **same** sign for both.
3. Calculate $X_{p1} = -R_0R_1/X_{s1}$ and $X_{p2} = -R_0R_2/X_{s2}$.
4. The parallel reactance of the **Tee** network is then $X_p = X_{p1} \parallel X_{p2} = X_{p1}X_{p2}/(X_{p1} + X_{p2})$.

This algorithm is very straightforward, but there are three problems:

- It assumes that the load is completely resistive.
- It doesn't start from a specification of the *inductance*, which is what we really want.
- It assumes a *lossless* inductor, and the capacitance values found will be slightly incorrect.

We deal with these problems like this:

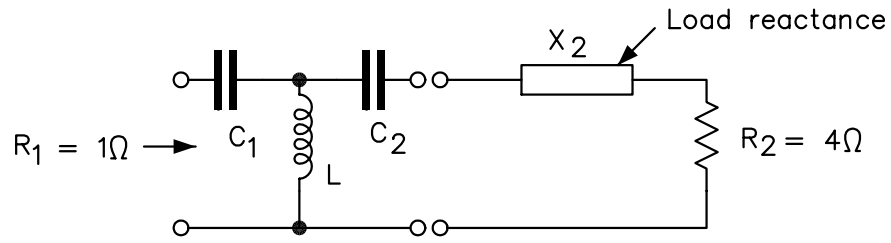

Figure 2: Adjusting for load reactance by compensating for it in the output capacitance reactance.

Load Reactance:

Load reactance, if any, is compensated for by subsuming its value into that of the output capacitor, C_2. Representing the load by its series equivalent circuit, we get figure 2:

Then

$$X_{C2} = X_{s2} - X_2 \quad (1)$$

$$\text{where } X_{C2} = \text{the final required capacitor reactance,} \quad (2)$$

$$X_{s2} = \text{the reactance value computed by the algorithm above,} \quad (3)$$

$$X_2 = \text{the load reactance (positive or negative.)} \quad (4)$$

Finding The inductance value:

This, (actually its reactance) only comes out in the last line of the algorithm, and is non-linearly related to R_o. Hence `teedesign` uses a binary bisection inversion algorithm to iterate to the value of R_o which gives the requested inductance value. If you open the source code in the `Octave` editor, this inversion starts at line 49.

Compensating for the lossy Inductor:

The lossy inductor effectively injects resistance in series with the "lossless" inductor assumed in the design algorithm, and this changes the input impedance. `teedesign` uses the wonderful Nelder-Mead simplex algorithm[2] `fminsearch` to adjust the capacitor values. The last line in the "expanded" dialogue of my program shows the number of iterations performed by this algorithm. If you set the parameter "`plotflag`" to 1 instead of zero near the start, you'll see a plot of the iteration procedure.

Estimating the Power Lost:

The approximate estimate of the power loss (shown only in the "expanded" dialogue) is made by a perturbation method. It assumes that the current through the inductor is unaltered by the insertion of the inductor loss resistance, and calculates the power in this resistance. The equation is

$$P_{lost} = \frac{P_{in}}{Q_L}\left[\sqrt{\frac{R_1}{R_0} - 1} + \sqrt{\frac{R_2}{R_0} - 1}\right] \quad (5)$$

[2]Search on "Nelder-mead" with Google for an explanation of how `fminsearch` works.

$$\text{where } P_{in} = \text{the input power (here assumed to be 100 Watt} \quad (6)$$
$$Q = \text{the } Q \text{ of the inductor,} \quad (7)$$
$$R_1, R_2 = \text{the source and load resistances,} \quad (8)$$
$$R_o = \text{the characteristic impedance.} \quad (9)$$

The "true" power loss is later calculated using matrix methods, in the source code portion starting at line 135. For an expanded explanation of these matching equations, see chapter 1 of our text.[3]

A Bug in the Program!

For some reason, the Nelder-Meade algorithm apparently fails here for some extreme values of load resistance. This may be because of a mistake in my code's implementation, but if so, I can't find it. If you can, tell me.

For example, if you choose the load resistance as 5000 Ω, this algorithm converges, but converges to incorrect capacitor values, which the final test rejects. Here's the dialogue:

```
Load resistance?  (50) 5000
Load reactance?   (zero)
Design frequency, MHz? (3.5)
Inductor Q?  (100)
Expanded results?  (1 for yes, return for no)

Minimum inductance is  22.96 uH
Inductance, uH?  (Minimum value)
Adjusted Zin =  55.13 +   0.19j
Load voltage    = 640.63 Volt rms
Power in load   =  82.08 Watt (100 Watt in)
True Power loss =  17.92 Watt

For matching at 3.50 MHz,
    C1 =  89.01 pF  (Lossless  88.80 pf)
    C2 =  37.41 pF  (Lossless  37.47 pf)
     L =  22.96 uH

  Peak Vc1        = 926.66 Volt
  Peak Vc2        = 220.23 Volt
  Peak Vinductor  = 932.38 Volt

  But adjusted Zin =  55.13 +   0.19j
 fminsearch failed convergence!  Increase L!
>>
```

Even if you *do* increase L, a 50 Ω match can't be found. However Kevin Schmidt's applet[4] *does* find a match using the "autotune" button, with $C_1 = 95.7\,\text{pF}$, $C_2 = 86.5\,\text{pF}$, $L = 21.6\,\mu\text{H}$, which is verified as correct by `LTspice`.

[3] Gary E.J. Bold and Sze M. Tan, "*Theoretical and Computer Analysis of Systems and Networks" obtainable from the University of Auckland Bookshop.*

[4] See http://fermi.la.asu.edu/w9cf/tuner/tuner.html