

User Guide to RealTime Match

Stephan Weber

Carl-Orff-Str. 9, 83052 Bruckmühl, Germany

E-mail: weberconnect@freenet.de

Standard Program Usage (Black-Box Mode).....	5
A Simple Network Design.....	7
A More Complex Network Design.....	10
Typical Next Steps in Matching Network Design	11
Advanced Program Usage (White-Box Mode)	12
A Simple Example.....	12
A More Complex Example.....	14
Purely Mathematical Examples.....	18
Inspecting the Log Files.....	20
Global Optimization	20

The application RealTime Match (ANPASS.EXE) has been created primarily for the synthesis of matching networks. However, there are also other calculations included, e.g., to design resistive damping elements or to find the optimum load impedance for power amplifiers (Class-A and Class-E). Program hints are available as bubble help and under the menu “Help.”

A further reason to create the program was to give designers more insights into optimization. In modern design environments, you can do optimizations, but with real circuits and real circuit simulations optimization is definitely time-consuming. If you trim a pot on a printed circuit board, you can get a feeling how the circuit reacts to your tweaks; this is not possible unfortunately in IC design (beside very trivial cases), but with RealTime Match you can learn about optimization quickly and then apply the know-how to your advantage in more complex, more realistic examples! Actually, the BFGS optimizer used internally is also available in many design environments such as ADE GXL, and it is often among the best in many numerical benchmarks!

Actually, it is also quite difficult to learn about optimization in an IC design environment, because the vendors just do not want to tell about the “tricks” used by the optimizer, or how the goal function is

composed. This RealTime app is much more transparent and provides several plots showing, e.g., how the optimization progresses, and a detailed log file is available too.

Some of the calculations in RealTime Match are straight forward; e.g., for the optimum load impedance, we can use the formula $R_{Lopt} \approx V_p^2 / 2P_{Wanted} \approx (V_{CC} - V_{sat})^2 / 2P_{Wanted}$, which is pretty accurate for class-A operation.

The screenshot shows the RealTime Match software interface. The top toolbar includes icons for file operations, simulation, and optimization. The main window is divided into several sections:

- Input:** Contains a "Switch" button, "Q definition" options, and input fields for Resistance R1 (50.000 Ω), Resistance R2 (1.00k Ω), and Frequency fo (1.00G Hz). There are also "Circuit Hints" and "Guide" buttons.
- Output:** Contains fields for Bandwidth, v=V2/V1, 20-log(v), Loss, and QBtot.
- Input/Output:** Displays circuit diagrams and calculated values for Supply Voltage V_{CC} (5.000 V), Saturation Voltage V_{sat} (500.00m V), Output Power P (200m W), Level (23.010 dBm), Io (88.89m A), and various R_{Lopt} values (R_{Lopt1}: 10.000 Ω, R_{Lopt2}: 50.630 Ω, R_{Lopt3}: 40.000 Ω, R_{Lopt4}: 202.500 Ω).
- Standard Series:** Lists standard resistor series (E6, E12, E24, E48, E96) with their respective percentage tolerances. There are buttons for "OK", "Exit", "Help", "Take Entries", and "Take and Optimize".

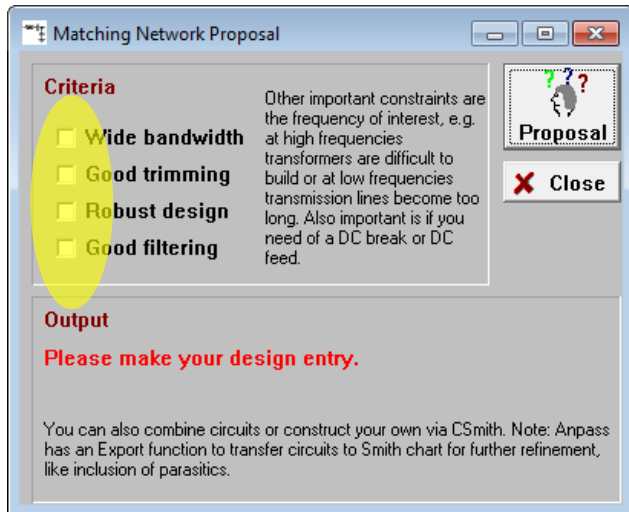
The bottom status bar shows the current tab as "RLopt" and a message: "Copy window contents to clipboard, achieved fmin= 0.0000".

RealTime with calculation for optimum load resistance (in this case the upper entries are not required)

This could be one part of a matching network design, and some networks can be also calculated directly (like those with two elements only). However, there are also matching networks which require (or at least have a benefit from) optimization techniques for their design.

The different calculations are available as different tabs, so first select a desired network (or calculation) type and then enter the design parameters, basically the design targets (like operating frequency and impedances) and further constraints. Note that some kind of matching networks requires either $R_G > R_L$ or vice versa, so you might be asked to toggle load and generator to fit to that constraint (use the "Switch" button).

If you hit the button “Info,” you will get some guidance on which kind of matching network is best suited for your requirements.



Get circuit hint from this dialog

You can use RealTime Match completely as a kind of black-box design tool; i.e., you can enter the design constraints, like the generator and load resistance and the frequency of operation—and just trust the results and look to frequency response. However, you can also inspect how the optimizer behaves behind the scenery.

We also include a little guide to optimization to support you on setting up an optimizer in true custom IC design environment. Actually, the choice of a suited optimizer is important, but by far not the only thing you should think of, e.g., goal definition and parameter setup is of similar importance. If you have done it few times, you usually get your own feeling for optimization quickly.

Guide for Optimization

Input

The two major design approaches are construction and optimization, often you mix them. How much construction makes sense and how much optimization depends highly on the designers capabilities, circuit type and spec difficulty. If your elements are near-ideal (like at low frequencies) and if you can easily over-design, then usually construction is leading to a ood solution. Which does not mean that no simulations and some tweaks are needed.

However, for difficult circuits (like RF) optimization can make a lot sense, e.g. it could happen that already one change in a parameter like the 'w' of the LNA transistor can change many performances (like NF, gain, input impedance, stability, bandwidth, etc.).

If optimization makes sense then the question is often which optimizer to use. Global optimizer tend to be more robust but slower. Local optimizer may stop early on a local optimum, which might exist in some cases or it starting point is bad.

Your primary constraints:

Number of parameters8

☒ Goal to parameter variation highly nonlinear

☒ Difficult correlations, many trade-offs

☐ Designer found a good starting point

☒ Starting point is probably bad

Output

Consider a global optimizer. You can expect significant improvements in app. 272 simulation points. Note if design is already close to optimum we would of course typically top earlier.

Close

Optimization guide built-in to RealTime Match

12.1 Standard Program Usage (Black-Box Mode)

Matching network design options:

Besides the impedances and operating frequency, the loaded Q and the bandwidth are most important. If you enter Q of zero, the bandwidth gets maximized, but not a true wide-band match is only possible for some of the configurations, like with the usage of transformers or via the 8-element LC band pass.

For simple band-pass circuits, there is a unique Q definition, and a fix relation between 3dB bandwidth and phase slope. For more complex networks, this is not the case anymore, so in ANPASS you can decide with a checkbox which Q definition you want to follow.

Graphical Windows:

Plots are available for the transfer function magnitude (in dB) and phase. Also a small Monte Carlo analysis is built-in to get a feeling for the sensitivity to component values. The component tolerances can be set in the main window, via button “Tol”—the default values are 5% (a uniform distribution is chosen, because fits usually quite well to discrete components—quite in opposite to the distributions of an IC process).

The axis scale can be adjusted via mouse click if you are close to end points of an axis. The origin can be set via double-click to the center of the plot. If you are not happy with your manual scale settings, then you can also click to the auto-scale entries in the menu.

Exporting graphics:

You can export the plots; three different file types are available: BMP, GIF, or JPEG. BMP is compatible with Paint program, but the file size is quite big. GIF is much more compact and also good for HTML integration. JPEG is also compact in size but has some loss in quality.

Window behavior:

The function “Always stay in front” from Help menu is very useful, especially for the graphical windows. In the input fields, you can use copy and paste as usual.

Misc:

RealTime Match includes the loss of the elements used by specifying a certain (common) element Q-factor, but parasitic inductances or capacitances are not included directly. A calculation for the effective inductance or capacitance is available under the tab “Leff/Ceff.

RealTime Match © S. Weber

File Edit Options Tools Help

Info $V(f)$ $\phi(f)$ Tol Optimization

Input

Q definition:
☒ Switch
☐ Q by $d\phi/d\omega$

Resistance R1: 50.000 Ω

Resistance R2: 350.000 Ω

Frequency f_0 : 2.40G Hz

Output

Bandwidth $\approx$: - Hz

$v=V_2/V_1$: -

20·log(v): - dB

Loss: - dB

Q_{Btot}: -

Input

Inductance L: 1.00n H

Capacitance C: 1.00p F

Resistance R_s: 100.00m Ω

Output

Ceff=1.29pF

Leff=1.29nH

$\omega L/R_s = 150.80$

$1/\omega C R_s = 663.15$

at $f_0=2.40$ GHz (input above)

Standard Series

☐ E6 Series (20%)

☒ E12 Series (10%)

☐ E24 Series (5%)

☐ E48 Series (2%)

☐ E96 Series (1%)

☐ E-Series for Export

Take Entries

Take and Optimize

OK

Exit

Help

LC Series Tank / LC Parallel Tank / Pi Section / T Section / Transformer / TRL / Passive L Section / Passive Pi/T Section / LC Balun / RLopt / Left/Ceff /

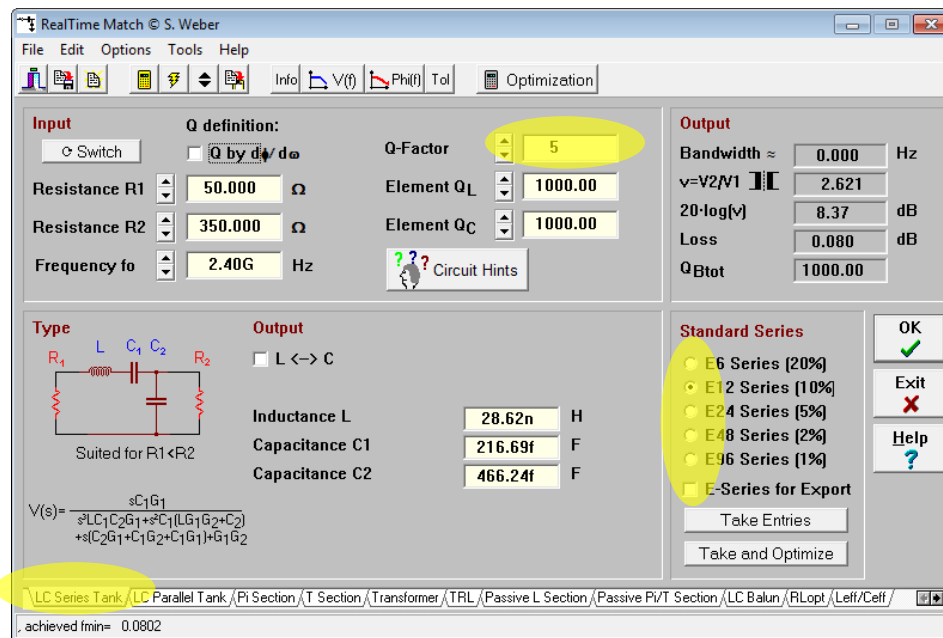
achieved fmin= 0.0226

Calculation of effective L and C

12.1.1 A Simple Network Design

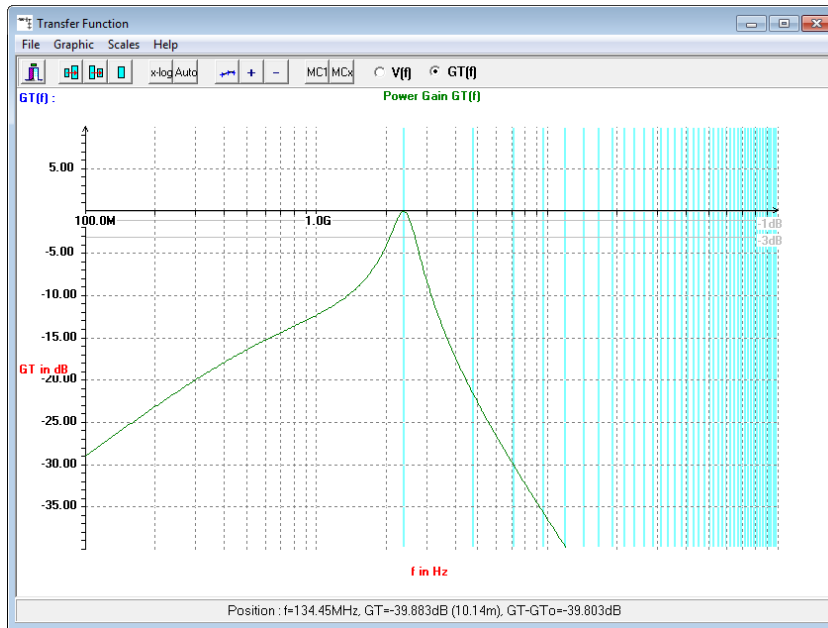
Imagine we need a matching network from $50\ \Omega$ to $350\ \Omega$. This is possible with a series LC tank circuit, so select this type in the tab row. The circuit consists of 3 elements, and for large bandwidth it can happen that one element effectively gets removed (like zero series inductance). By default, the circuit uses a parallel output capacitor and a series LC tank, but with the checkbox “L $\leftrightarrow$ C” you can also exchange all element types.

Disable the Q-type checkbox and set the Q-factor to 5 to get a narrow-band solution; the circuit will be calculated immediately and you can press “ $V(f)$ ” to get the frequency response plotted.



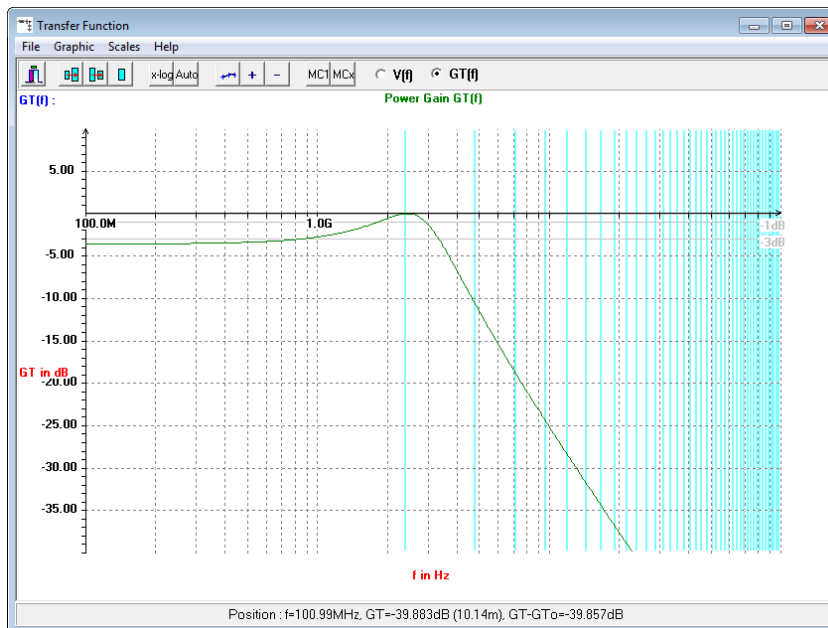
Series LC network for 24 GHz and narrow bandwidth

Hint: The calculated outputs are immediately available after your entries (no need to press OK or return). In addition, you can get the closest standard values (like 27 nH, 220 fF) via double click in the output field. The standard series can be set on the right side of the main window.



Plot for transfer function

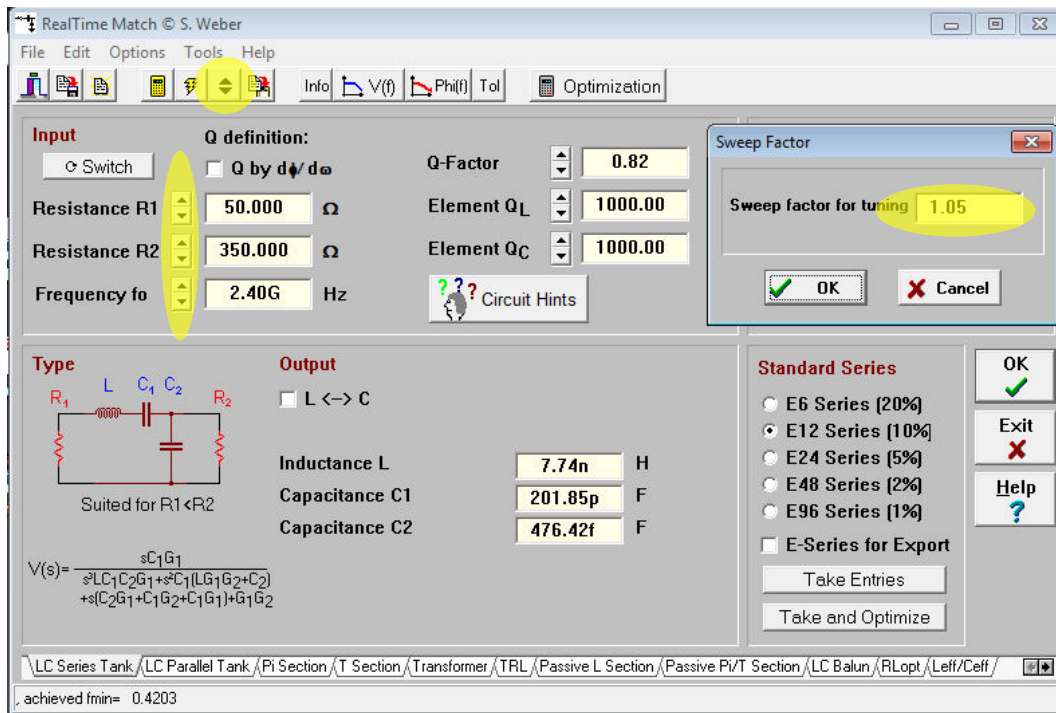
If you want a wideband solution, just set Q to zero and the series capacitor will become huge and indeed, the frequency response becomes now much wider. Note: Actually, there is a limit on bandwidth which depends on circuit type and the ratio between load and generator impedance. The more the extreme this ratio is, the smaller the achievable maximum bandwidth will be. The advantage of more complex circuits is that they usually offer more flexibility and also often a larger possible bandwidth.



Circuit optimized for maximum bandwidth

Note: The graph windows support using the mouse to read-out x,y values and we display lines for –1dB and –3dB.

Many input fields in the main window come with spin buttons aside; if you use it for generator impedance, you can watch in parallel to any output you want, e.g., to the element values in the output section and check their changes during the tweaks!

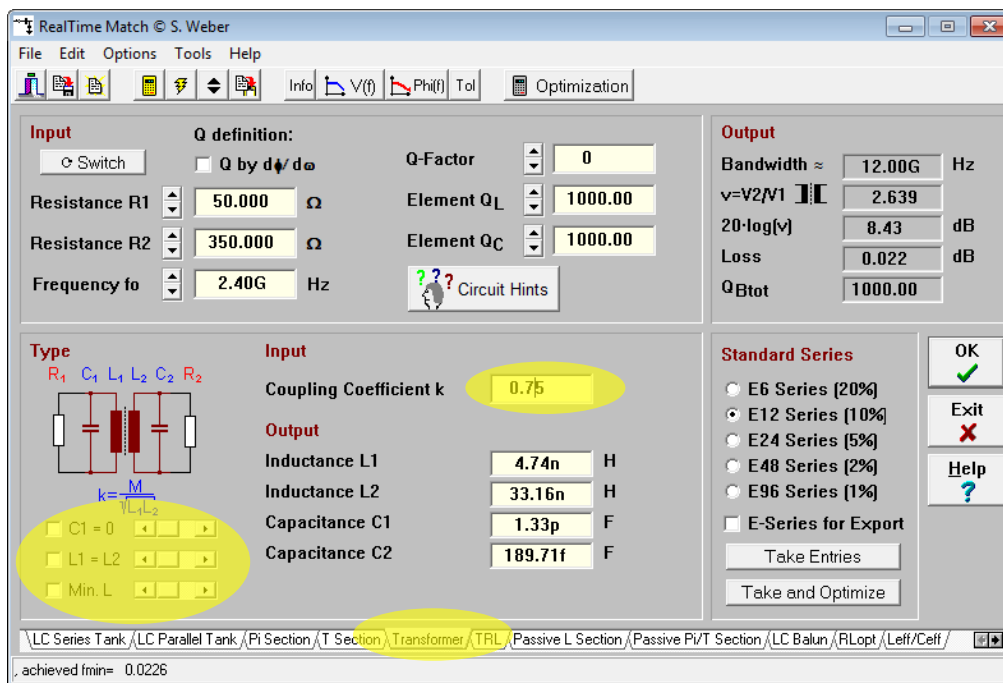


Sweeping mode in RealTime Match

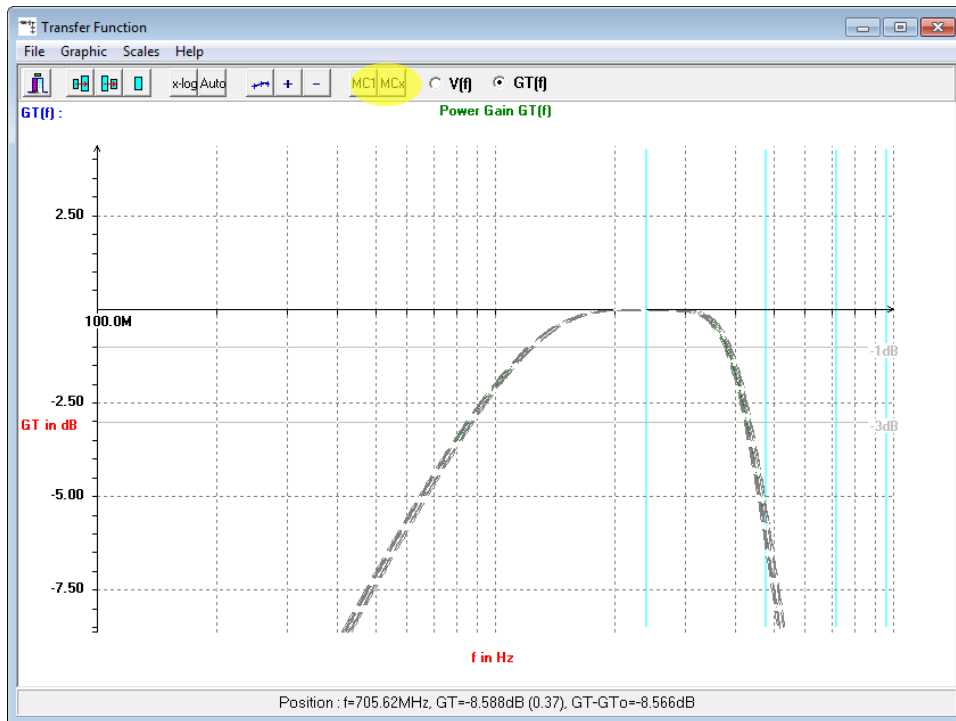
Note: Sweeping is also possible in the frequency plot windows. Just use the spin buttons in the main window; then, press Ctrl-A to update the plot.

12.1.2 A More Complex Network Design

With transformers you can design matching networks in a quite flexible way; e.g., a wideband match is possible if you are able to provide a transformer with high coupling factor k . For a good power match, we also need parallel capacitors to cancel the inductance of the transformer. So all-in-all, we need to take into account the 2 capacitors and the 3 parameters defining the transformer (primary and secondary inductance plus the coupling factor), so we have 5 parameters in total. This gives quite some freedom, so to achieve a certain operating center frequency and bandwidth, we may also give further constraints like whether we want to avoid an input or output parallel capacitor, by setting sliders in the GUI accordingly. On the other hand, not for all k -factors a real good solution is possible. The big advantage when using RealTime Match is that you can immediately check how certain constraints impact the matching network behavior.



Matching network with transformer ($k = 0.75$)



Frequency response with MC family plot (zoom used)

Press the MC button a MC plot. To see the details better zoom-in using the mouse.

12.1.3 Typical Next Steps in Matching Network Design

After the synthesis with near-ideal elements via RealTime Match, you may create a schematic and use real components. Usually, the behavior will be slightly worse, so you may use a Smith chart or an optimizer to get back the near-ideal behavior (if possible). This is typically also required if you add, e.g., bond wires and pads. Also RealTime Match is only designed to transform from a real-valued generator impedance to a real-valued load impedance, so to cancel imaginary parts you may need to add or extend the matching network further (or try to retune it for complex impedances).

Also note that RealTime Match optimizes the circuit for maximum flat bandwidth if we want a wide-band match, which is equivalent to Butterworth behavior, but a Chebyshev filter could give a larger bandwidth at the cost of some pass-band ripple. An optimizer can usually easily tweak a Butterworth filter to get another filter type, like Chebyshev or linear-phase filters —just the optimization criteria have to be adopted!

Note: The element Q-factor for L's and C's can be already set in ANPASS. It could happen that too low Q introduces significant damping and less sharp filter characteristics. This is a normal effect, but how strong it is varies quite a lot from circuit to circuit.

12.2 Advanced Program Usage (White-Box Mode)

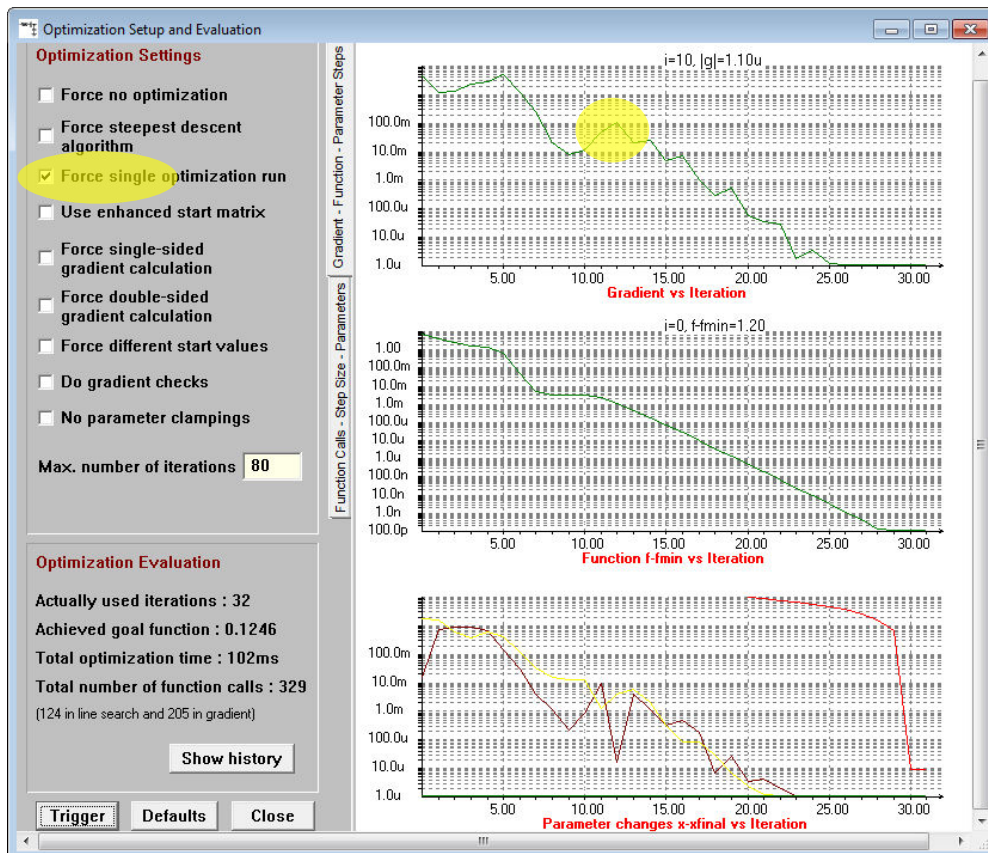
As mentioned behind the scenery, a BFGS optimizer is designing the different circuits, with up to 8 elements. The calculation of the frequency response is done internally by using analytical formulas, which have been derived by Symbolic Spice. You find the formulas in the main window for each network type.

The key parameters of such an optimization run can be inspected by clicking to the “Optimization” button. For instance, you can see how many BFGS iterations have been executed, and how many simulations have been required. In addition, plots are provided to show the gradient and the behavior of all parameters during the optimization.

Actually by default BFGS is called two times, to solve difficult problems, so by default you see the results of the second optimization as outputs in the main windows and also in the optimization history window.

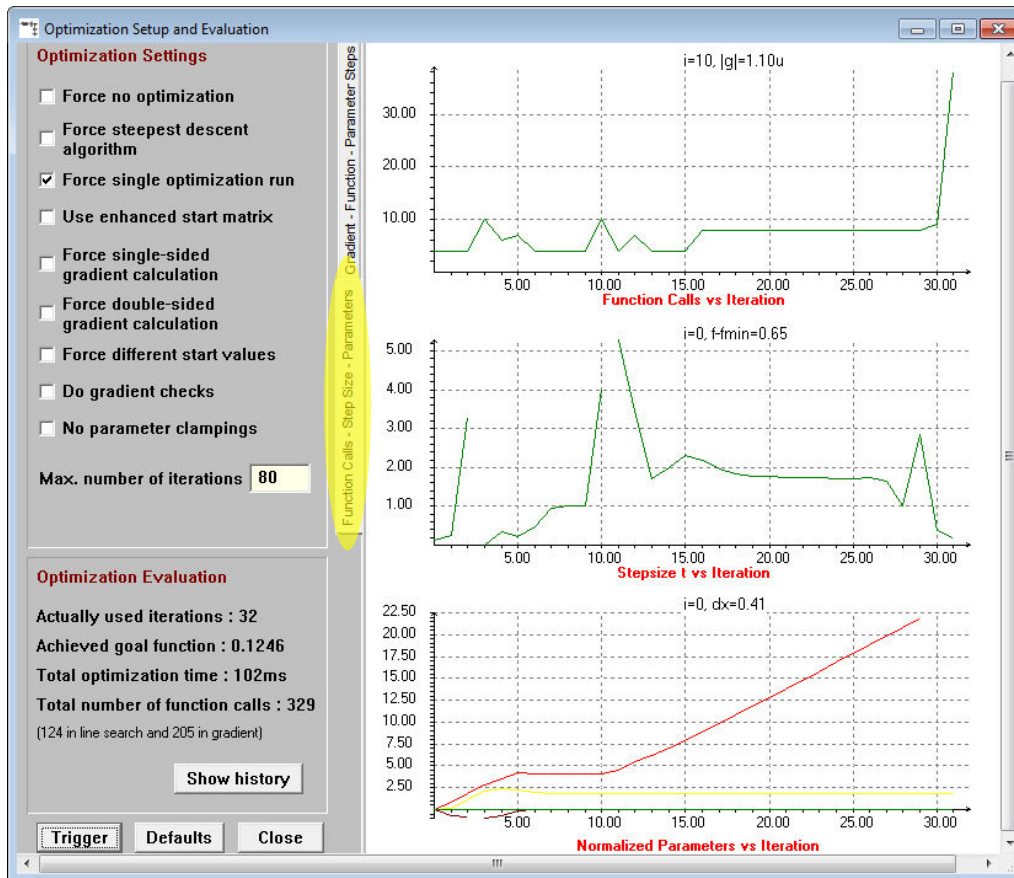
12.2.1 A Simple Example

The simplest use model is to run a matching network design as usual (like for the series LC circuit), and then to press “Optimization” to inspect the BFGS execution.



Optimization window (select Force single run)

At the 1st page, you can see plots vs iteration for the gradient, for the goal function, and for the parameters to be optimized. The goal function drops monotonously as (highly) desirable for an optimizer. However, the gradient is not always monotonous. That lives, e.g., around iteration 10 g even starts to grow significantly.



Switching to the 2nd page

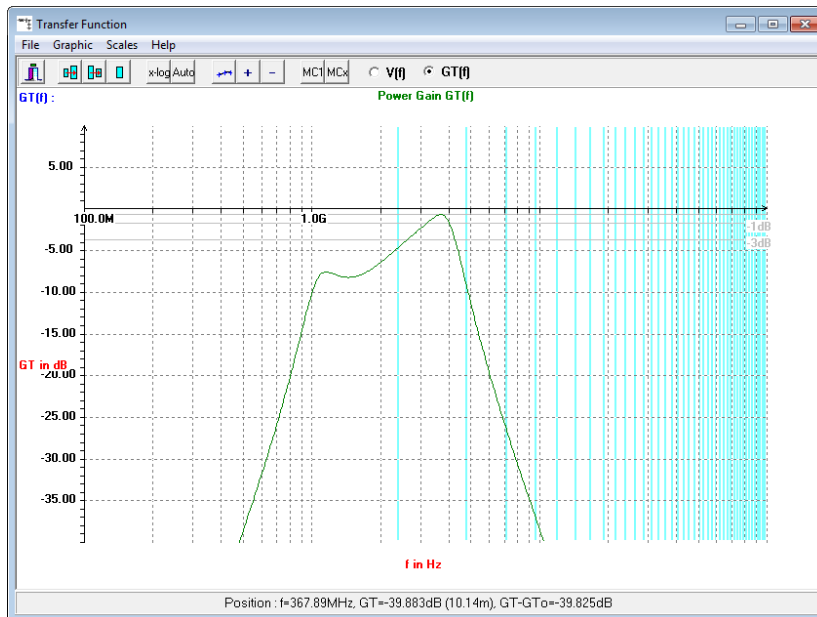
The 2nd page shows an interesting behavior for one of the optimized components. It is rising like a ramp, because actually the optimum value is infinite. The ramp looks slow, but actually we treat the parameters logarithmically, so the ramp is exponential—so not slow at all.

Interesting is also the step-size parameter plot. Ideally, it is close to unity, not in the case of difficult optimizations.

12.2.2 A More Complex Example

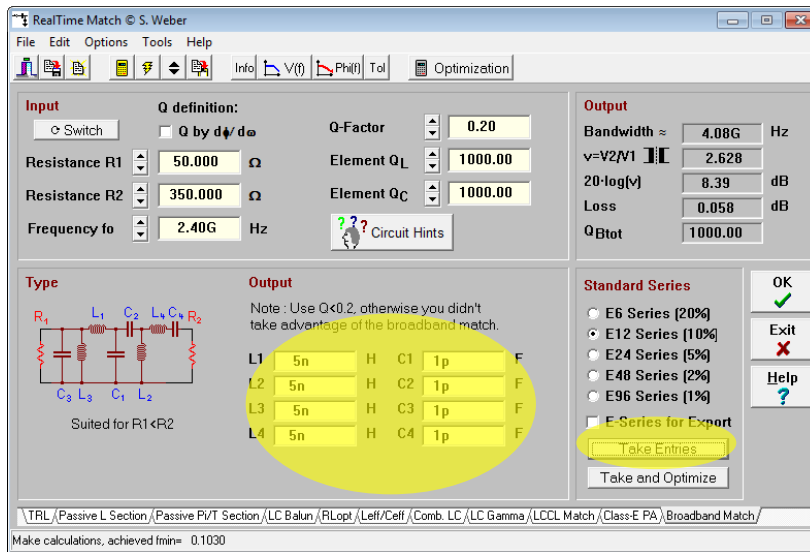
You can also do more complex experiments; e.g., it is possible 1st to enter the matching network elements directly in the main window, and let the optimizer start from such user entries. This way you can get a feeling how BFGS behaves if the starting point is difficult. To inspect how bad the initial set of element values is press the button “Take values” and plot, e.g., “ $V(f)$.” For a complex matching network like the 8-element band-pass, it is very difficult to provide good start values by hand. You may try to yourself a design via Smith chart, and then enter the values in RealTime Match.

Or you may enter just 5nH for all inductors and 1pF for all capacitors.



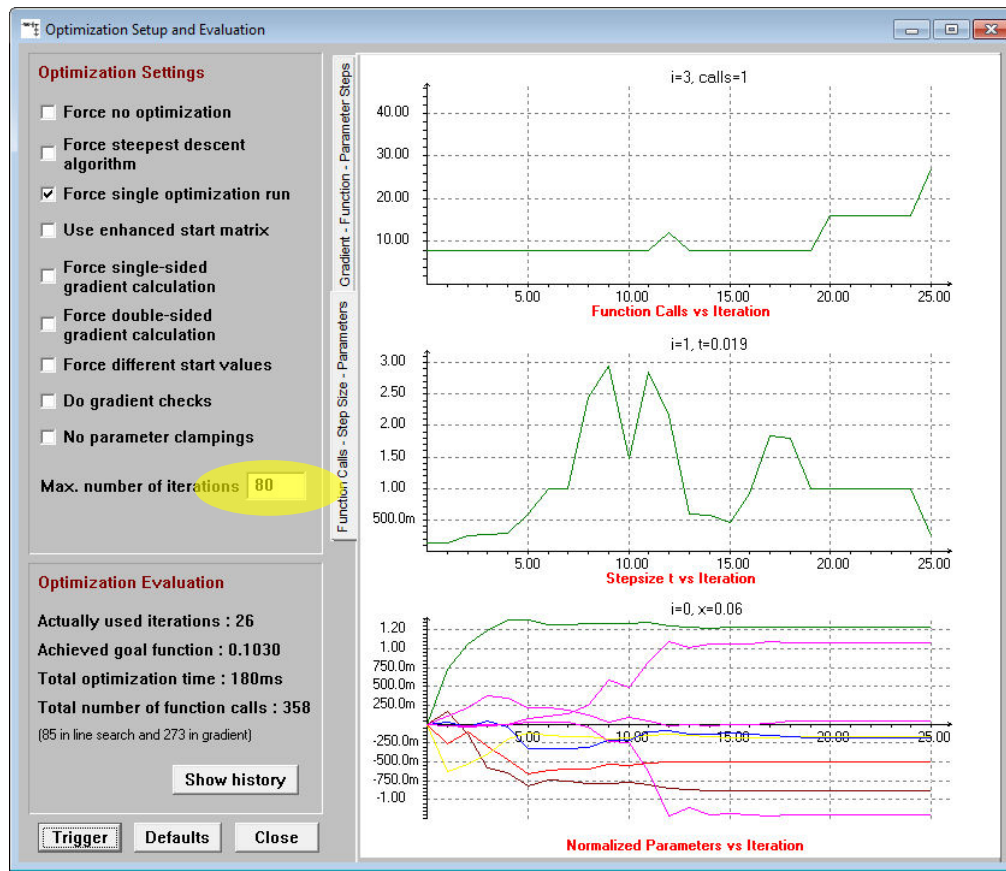
Circuit with manually entered component values

Note: In the optimization setup window, you can also disable the optimization completely, so we get the circuit and the performances with element values from built-in manually hard-coded values. In addition, there is a checkbox for multiple starting points—which we will use at the end of this chapter.



Frequency response and circuit with manually set start values

If you press now "Take and Optimize," the optimizer starts with your user entries, and the optimization might be quite difficult. However, BFGS is quite powerful and still takes only 26 iterations only (only few more as from the built-in starting points). The parameters have been adjusted by app. 1:3, as observable at the 2nd page.



Optimization history with manually set start values.

Beside the BFGS result window, also log files are available and those can be inspected with a text editor. In the log files, we can save information such as the current set of (normalized) parameters, the gradient, the Hessian matrix, and its eigenvalues. This way you can see many more details how difficult the optimization was.

Note: After app. 15 iterations, the circuit is at least almost optimized, like 99%. With real Spice simulations you may stop earlier to save some time. In ANPASS, you can do this by limiting the number of iterations. In the current optimization, this value is at 80 (=default), but BFGS stops early because the gradient is below our (internally set) target.

A general question when doing optimization is whether the solution is unique. This can be checked by running multiple optimizations with different starting points. This feature is available just by setting a checkbox. Usually RealTime Match works very accurate, so that the different solutions show only minor variations (see log file created in the directly of the app itself). This might not be the case if more numerical noise is present like from a transient analysis, or if you select single-side gradients (see figure).

Parameter uncertainty estimation :

dX1 : 0.00186%

dX2 : 0.00234%

dX3 : 0.00228%

dX4 : 0.00279%

dX5 : 0.00574%

dX6 : 0.00600%

dX7 : 0.00995%

dX8 : 0.00905%

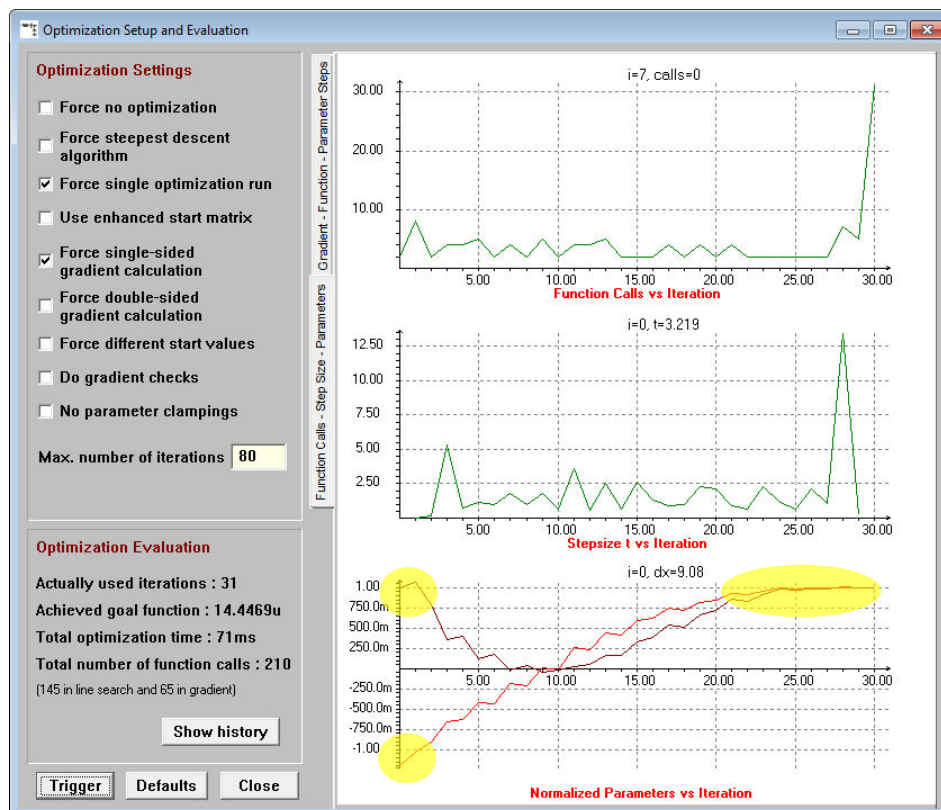
f at achieved optimum : 0.102995606, 0.102995606, 0.102995606, 0.102995605, 0.102995606,
0.102995606, 0.102995606, 0.102995606

End of the log file if using multiple starting points

12.2.3 Purely Mathematical Examples

As mentioned, some calculations do not require optimization techniques, so behind those tabs we put some classical mathematical optimization examples like the minimization of the Rosenbrock “banana” function or a pure quadratic function. The Rosenbrock function has only two variables, but is quite difficult to optimize, so it is used in many benchmarks. The interesting part in the optimization of a pure quadratic function is to check how accurate BFGS can approximate the inverse Hessian matrix and if BFGS really terminates after n (=number of variables) steps.

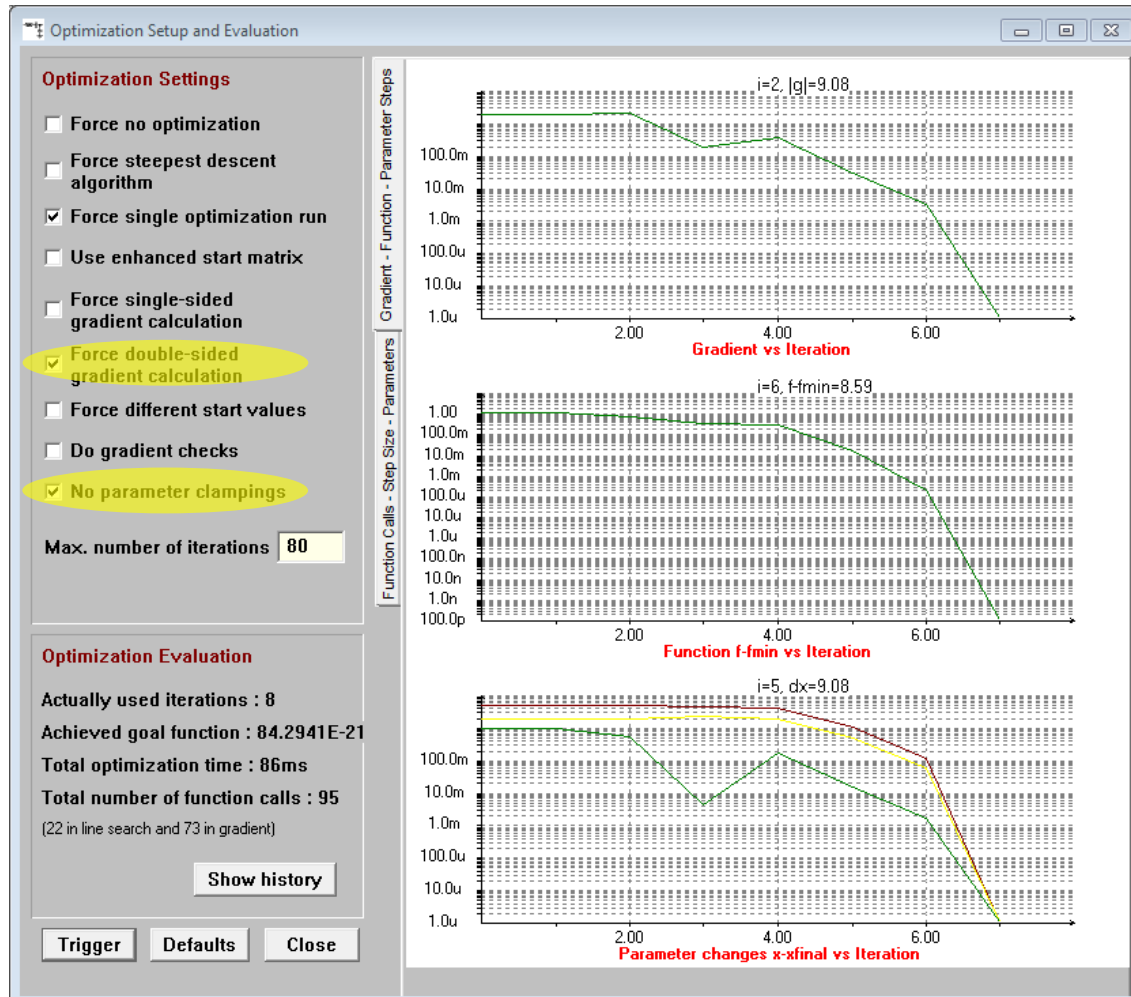
The Rosenbrock function will be optimized if you select the tab “RLOpt” and then click at “Optimize.”



BFGS on the Rosenbrock “Banana” function

We use the classical starting point $(-1.2, 1)$ and the optimum $(1, 1)$ is reached after 31 iterations and 210 function calls. This is the case if we use single-sided gradient, and here we stop at $f = 14\text{E}-6$. Using double-sided gradients we would need few more points, but could achieve a significantly lower goal function value. With steepest gradient, we would need much more than 200 iterations!

A 4-parameter quadratic function will be optimized when selecting “TRL” tab. In a good setup, we can indeed optimize it in an almost perfect way (error function below $1\text{E-}19$ in only 6 iterations)!



BFGS on a 4-variable quadratic function

Note that no parameter clamping is checked, so we allow big parameter steps. The reason is that BFGS has an automatic internal step-size setting and is able to make big steps forward to the solution. This is the better setting for such pure quadratic problems, because BFGS can optimize those from (almost) any starting point you want! One reason to take a little more iterations than expected from pure theory is that the gradient accuracy and linear search accuracy are limited.

Inspecting the Log Files

In addition to the optimizer output directly available in the user interface, also log files get saved providing additional information. With them you can see in detail which way the optimizer has taken or what is the Hessian matrix, its eigenvalues, etc.

Two sets of such log files have been provided with the app showing the optimization of the 2-variable Rosenbrock function and for the LC band pass. For the first you can easily create a 2D plot in Excel, whereas optimization on more than two variables is harder to visualize. You may pick sets of two variables in the 8-parameter filter optimization to get at least some understanding.

12.3 Global Optimization

Internally, a BFGS optimizer is used for all the electrical examples. Actually some have local minima, but still BFGS is reliable enough if the starting point is not too bad. For extreme problems with many local minima, BFGS would fail, and the traveling salesman problem is one example. We include it to the RealTime app and solve these optimization problems via simulated annealing.

In the Internet, a lot of literature is available on both the simulated annealing and the traveling salesman problem. Many articles are very informative and nice to read. In our app, you can run either one optimization at the time, or many of them to get also some statistical behavior of the optimizer (like how much difference is between the best and the worst optimization