S-26.3120 Radio Engineering, laboratory course

Lab 4: Amplifier

Pre-study Report

March 17, 2014

Tuomas Leinonen, 84695P, Group 1

1 Theoretical background

Answer the following questions briefly.

For more thorough discussions, the reader is encouraged to get acquainted with the reference material listed at the end of this document.

1.1 Describe the procedure to design a low-noise microwave transistor amplifier.

First of all, one needs some "goals"; a set specifications to be fulfilled. These are given (like in our labs) or derived from the system specifications when other blocks are taken into account. These are usually given in terms of (stability), power gain, bandwidth and noise; along with physical properties of the application and/or usage environment (temperature, size, price/cost and biasing/power constrictions, for example).

Next, the PCB-material and transistor(s) and its (their) configuration, including DC biasing, is selected carefully based on specifications. It is best to go for the simplest design that meets the requirements by a suitable margin. After this, transistor loading is determined. That is, the source and load reflection coefficients (i.e., the matching circuits, see question 1.5) yielding wanted properties (listed in the previous paragraph) are found. For this, one commonly uses hand calculation with S-parameters, Smith-charts and possibly Matlab, or circuit simulators along with other CAD-tools.

Now that we have the "ideal" design, we use a RF-simulator, such as ADS, to design the actual amplifier layout. This icludes the biasing and matching circuits as accurately as possible. The design is then rigorously simulated, tweaking the circuit until optimum performance is achieved. One should take practical limitations into account in this phase.

The amplifier is then manufactured and measured in real-life. If the specifications are not met, designers must seek to discover the source of the problem. Fixing the problem may require some tweaking in the already manufactured product or a complete redesign depending on its severity. Careful layout design and rigorous simulation will help to avoid such situations.

1.2 Describe the various methods by which the amplifier can be stabilized. What are the tradeoffs when using different methods to stabilize the transistor?

Using the two-port approach, an amplifier may oscillate when the either of its ports, input or output, presents a negative resistance. This corresponds to a situation where $|\rho_{\text{in}}| > 1$ or $|\rho_{\text{out}}| > 1$. Thus to stabilize a resistor, these are to be made less than unity. One should note that said reflection coefficients are dependent on input and source loading (i.e., $\rho_{\text{in}} > 1$ and ρ_{L}), and that stability is frequency dependent due to its S-parameters being a function of frequency. Thus the designer needs to confirm that the circuit is stable at all frequencies.

This may be achieved by atleast two means; by resistive loading or by adding negative feedback. Both of these methods are mainly used for stabilizing broadband amplifiers, and the

stability of narrowband amplifiers is ensured by a careful selection of source and load reflection coefficients. This is because all of the aforementioned stabilization techniques have negative effect on gain, noise and/or *VSWR* performance of the amplifier. Especially resistive loading at source has a drastic effect on noise performance. On the other hand, stabilizations methods may in some cases be used to meet design goals.

1.3 What is the difference between the unilateral and bilateral scenario in designing transistor amplifiers and how does it impact the design? What is the unilateral figure of merit?

A two-port is said to be unilateral ("one-directional") if and only if its $S_{12} = 0$. Otherwise the two-port in question is bilateral ("two-directional"), and bilateral design routine must be used. While bilateral design techniques may be used always, unilateral techniques are easier than their bilateral counterparts. This is due to the source and load reflection coefficients being independent of each other and may thus be designed separately. The loading of bilateral amplifiers is rather complicated since both source and load matching (ρ_{in} and ρ_{out}, respectively) are cross-dependent. This may be seen in the definition of said reflection coefficients:

$$\rho_{\text{in}} = S_{11} + \frac{S_{12}S_{21}\rho_{\text{L}}}{1 - S_{22}\rho_{\text{L}}} = S_{11}, \text{ when } S_{12} = 0, \tag{1}$$

$$\rho_{\text{out}} = S_{22} + \frac{S_{12}S_{21}\rho_{\text{S}}}{1 - S_{11}\rho_{\text{S}}} = S_{22}, \text{ when } S_{12} = 0. \tag{2}$$

Because of this, most textbooks (like [3]) present "advanced" design procedures, like constant gain and noise circles, only of the unilater case. Luckily a bilateral amplifier can be designed using the unilateral methods if its $|S_{12}|$ is negligibly small. The validity of this $S_{12} \approx 0$ approximation may be evaluated by calculating *the unilateral figure of merit* U for the amplifier in question. It is defined as follows:

$$U = \frac{|S_{11}||S_{12}||S_{21}||S_2|}{(1 - |S_{11}|^2)(1 - |S_{22}|^2)}. \tag{3}$$

When designing amplifier of maximum gain using (simultaneous) conjugate matching, maximum transducer power gain error caused by unilateral approach $G_{\text{T}}/G_{\text{TU}}$ is given as

$$\frac{1}{(1+U)^2} < \frac{G_{\text{T}}}{G_{\text{TU}}} < \frac{1}{(1-U)^2}. \tag{4}$$

Depending on the specifications, an error of few tenths of a decibel usually justifies the use of unilateral design procedure.

1.4 What is the difference between the operating and available power gain circles? Which of these should be considered for designing an LNA?

In literature, one may find three different power gain definitions for a two-port. In [3] they are defined as follows (one should note that different authors use slightly different, yet still a tad confusing, naming convention):

- Operating power gain G_P, shown in Eq. (5), is the ratio of power dissipated in the load Z_L to the power delivered to the input of the two-port network. This gain is independent of Z_S, although the characteristics of some active devices may be dependent on Z_S.
- Available power gain G_A, shown in Eq. (6), is the ratio of the power available from the two-port network to the power available from the source. This assumes conjugate matching of both the source and the load, and depends on Z_S, but not Z_L.
- Transducer power gain G_T, shown in Eq. (7), is the ratio of the power delivered to the load to the power available from the source. This depends on both Z_S and Z_L.

While transducer gain is the most useful in general, we need to compare the first two to each other in this task.

$$G_\mathrm{P} = \frac{P_\mathrm{L}}{P_\mathrm{in}} = \frac{|S_{21}|^2(1-|\rho_\mathrm{L}|^2)}{(1-|\rho_\mathrm{in}|^2)|1-S_{22}\rho_\mathrm{L}|^2} \tag{5}$$

$$G_\mathrm{A} = \frac{P_\mathrm{avn}}{P_\mathrm{avs}} = \frac{|S_{21}|^2(1-|\rho_\mathrm{S}|^2)}{|1-S_{11}\rho_\mathrm{S}|^2(1-|\rho_\mathrm{out}|^2)} \tag{6}$$

$$G_\mathrm{T} = \frac{P_\mathrm{L}}{P_\mathrm{avs}} = \frac{|S_{21}|^2(1-|\rho_\mathrm{S}|^2)(1-|\rho_\mathrm{L}|^2)}{|1-\rho_\mathrm{S}\rho_\mathrm{in}|^2|1-S_{22}\rho_\mathrm{L}|^2} \tag{7}$$

Gonzales recommends a procedure based on constant operating power gain for practical designs due to its simplicity, saying it's common practice when S_{12} cannot be neglected. On the other hand, using G_A is beneficial since it can be plot on the same Smith chart as constant noise circles, both being dependent on $|\rho_\mathrm{S}|$. This way one may find a suitable compromise between gain and noise, and is thus more suitable for this lab exercise.

Pozar uses only unilateral approach, looking at source and load "matching gains" separately. Constant noise circles may be plot on the same Smith chart as constant source matching gain circles.

1.5 What are the different types of matching networks that can be used for matching the source and load? Enumerate the advantages/disadvantages of the different methods.

In general, matching circuits may be divided into two categories: lossy and lossless matching circuits. The former method uses attenuators whereas the latter is accomplished using lumped

inductors and capacitors, or microstrip transmission lines. Most notable pros and cons of each realization are listed below [1].

Attenuation:

- \+ Inherently wideband
- \+ Small size
- − Lossy and noisy
- − Cannot realize arbitrary ρ or Z

Microstrip:

- \+ Well characterized and flexibility in design
- \+ Low loss and high performance
- \+ Has better harmonic tuning capability for high-efficiency applications
- − Strong interaction effects between elements and discontinuities in packed design
- − Large in size even with interactions included

Lumped *LC*-network:

- \+ Capable of transforming high impedance ratios
- \+ Minimum interactions between elements due to small size
- \+ Compact in size
- − Low Q, limited DC power handling capability
- − Substandard performance due to low Q at the output of a power amplifier in terms of P_{out} and PAE

LC-networks are used at low frequencies by placing one or more series or shunt elemements after one another. When using two or less elements, one may find the matching circuit analytically or by using a Smith chart. Inductors may also be used for transistor matching when used as degeneration elements. IDCS refers to inductively degerated common source, a topology where series inductances are placed at gate and source. These interact with the parasitic gate-source capacitance forming a resonator.

Microstrips may be used by alternating impedance along the same line (e.g., tapering, quarter-wave transformers or so-called "high/low-Z_0 matching"), or by using separate (series/)parallel stubs of certain, distance (from load), length and Z_0 (e.g., single or two-stub matching). One might also combine aforementioned approaches, or use baluns.

For our amplifier design, the microstrip based approaches are recommended. The matching circuits of the amplifier may be found simply by using the traditional conjugate-approach of the selected method to match ρ_{S}^* and ρ_{L}^* (looking toward source and load, respectively) to Z_0.

1.6 How is the reflection coefficient of the designed amplifier calculated? What is the compromise when designing an amplifier for low-noise and maximum gain operation?

The input and output reflection coefficients ρ_{in} and ρ_{out} of any two-port are found using Eq. (1) and (2), respectively. One should note the cross-depency in the bilateral case (i.e., when $|S_{12}| > 0$). Source and load reflection coefficients are chosen based on wanted amplifier characteristics (i.e., gain, return loss, noise performance, linearity, optimum power and PAE), that are often contradictory to one another. Here we focus only on gain and noise figure of the amplifier.

Using Pozar's terminology/concepts, we may only affect the "matching gains" of the transistors. These may be set quite freely within certain bounds (enforced by stability requirements, for example) using matching circuits to create suitable loading at the input and output. The gain due to $G_0 = |S_{21}|^2$ is constant at given frequency and used DC-biasing.

Maximum gain is obtained when simultaneous conjugate matching is used. Now $\rho_{\text{in}} = \rho_{\text{S}}^*$ and $\rho_{\text{out}} = \rho_{\text{L}}^*$, and both "matching gains" and return losses are maximized. This situation is rather easy to achieve in both uni- and bilateral cases, assuming the transistor remains stable. The obtained bandiwdth is usually quite limited, though.

Optimal noise performance is obtained by using suitable source mismatch to cancel some of the correlated noise. Doing so decreases the source matching gain. The output matching has no effect on noise, and may thus be conjugate-matched to achieve higher gain. One should note bandwidth requirements here as well.

Long story short: there's a trade-off between maximum gain and optimum noise performance. The designed may trade some of the source matching gain in favor for better noise performance. For this purpose, constant gain and constant noise figure circles are plot on the same Smith chart for ρ_{S}.

1.7 Briefly discuss the impact of grounding using plated-through holes (PTH) via.

Most commonly RF field-effect transistors, such as the pHEMT in question, are used in common-source (CS) configuration. In this configuration, gate and drain are used as input and output nodes, respectively. The source node is connected to signal ground as such or a through a so-called degeneration element; a resistor (rare) or an inductor (IDCS).

Using a degeneration element is justified when performance improvements are achieved through well-thought design. Careless design may lead to unstable operation, especially at higher frequencies due to parasitic effects. In practice, these parasitic RLC are present in all connections – also in this grounding, where L is typically dominant due to high operating frequencies. In some cases, the use of plated-through holes (PTH) is a non-negligible source of inductance, limiting the size of the "actual" degeneration-inductance if used. Especially when a thick substrate is used.

Two PTHs per a source connector (two source connections, four PTHs in total) are embedded in the provided S-parameters to help the design process. They assume certain substrate thickness and PTH placement.

2 Calculations

For this lab assingment, the design goals and other specifications are given in the following two tables (Tables 1 and 2).

Table 1: Design specifications for Group 1.

Paramater	Symbol	Value	Unit
Operating frequency	f_{op}	2.5	GHz
Drain-to-Source DC-voltage	V_{DS}	2	V
Drain-to-Source DC-current	I_{DS}	30	mA
Minimum gain	G_{min}	13	dB
Maximum noise figure	F_{max}	0.8	dB
Minimum return loss	RL_{min}	15	dB

Table 2: Transistor and PCB specifications.

Transistor:	Avago Technologies ATF-35143 (pHEMT)
PCB:	RT Duroid 5870 ($h = 0.787$ mm, $\epsilon_{\mathrm{r}} = 2.3$)

All calculations are based on a MATLAB script using only the so-called "core features" of MATLAB. This rather lengthy script is shown in Appendix A. For more comprehensive study, a real circuit simulator with layout capabilities should be used. Such simulations are left for latter part of this lab assignment.

2.1 For the given specifications of operation, calculate the stability factor and draw the input and output stability circles. Is the transistor unconditionally stable? If not, propose methods by which the stability can be improved.

The stability of the transistor was investigated using provided S-parameters over the whole frequency range $0.5 \ldots 18$ GHz. Both Rollet's $|\Delta| - K$ test and the μ test were used. The results of the μ test are shown in Fig. 1. The stabilty factors at the specified operating frequency of 2.5 GHz were found to be $|\Delta| = 0.419$, $K = 0.474$ and $\mu = 0.437$. Using the specified biasing point, the transistor is potentially unstable. Unconditional stability is achieved when $|\Delta| < 1$ and $K > 1$, or equivalently when $\mu > 1$.

For a potentially unstable transitor we may draw input and output stability circles. A two-port is stable, when conditions $|\rho_{\mathrm{in}}| < 1$ and $|\rho_{\mathrm{out}}| < 1$ are met simultaneously using passive loading $|\rho_{\mathrm{S,\,L}}| < 1$. For unilateral amplifiers, these conditions simplify to $|S_{11,\,22}| < 1$.

For bilateral amplifiers, stability circles are drawn for $|\rho_{\mathrm{in,\,out}}| = 1$ using Eq. (8-11). Output (input) stability circle is a circle with a radius R_{L} (R_{S}), centered at C_{L} (C_{S}). Values of ρ_{L} (ρ_{S})

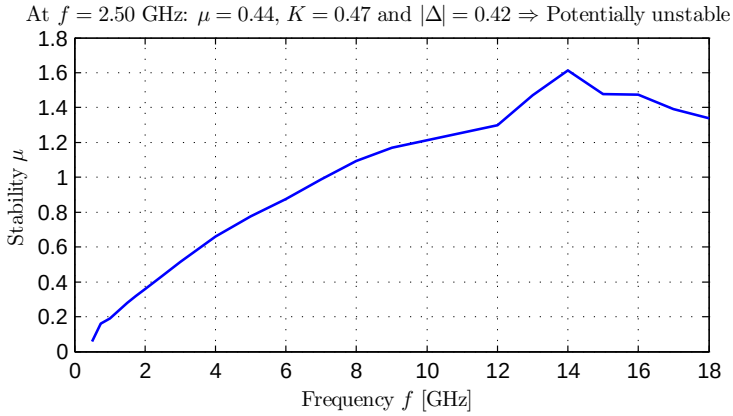

Figure 1: Results from the μ test: the transistor is unconditionally stable when $f > 7$ GHz.

correspond to a situation where $|\rho_{\text{in}}|$ ($|\rho_{\text{out}}|$) is unity. If $|S_{11,\,22}| < 1$, $|\rho_{\text{S,\,L}}| = 0$ (origin) is in the stable region of the Smith chart. Both input and output stability circles of the transistor in question are shown in Fig. 2. The (un)stable regions of the Smith chart are also labeled.

$$C_{\text{L}} = \frac{(S_{22} - \Delta S_{11}^*)^*}{|S_{22}|^2 - |\Delta|^2} \quad \text{(center)} \tag{8}$$

$$R_{\text{L}} = \left| \frac{S_{12}S_{21}}{|S_{22}|^2 - |\Delta|^2} \right| \quad \text{(radius)} \tag{9}$$

$$C_{\text{S}} = \frac{(S_{11} - \Delta S_{22}^*)^*}{|S_{11}|^2 - |\Delta|^2} \quad \text{(center)} \tag{10}$$

$$R_{\text{S}} = \left| \frac{S_{12}S_{21}}{|S_{11}|^2 - |\Delta|^2} \right| \quad \text{(radius)} \tag{11}$$

One might stabilize a potentially unstable amplifier using resistive loading, as was discussed in section 1.2. There are four ways the resistor may be connected to the amplifier, and all of these connections were studied separately. The study were two-fold, conducted using transmission parameters derived from provided S-parameters. First, resistance values yielding unconditionally stable circuit at $f = 2.5$ GHz were determined. These values were processed further to determine resistance values capable of meeting specifications. The results of this study are shown in Table 3.

For the rest of this report, the transistor is assumed to be stabilized using a series resistance of 100 Ω connected to the output (drain) of the transistor. This stabilization method and the value was selected because it was found to meet all requirements with relatively good margins. The value of 100 Ω is very good choice due to its availability in all standardized resistor series (E6...192). A more thorough discussion is presented in section 2.3.

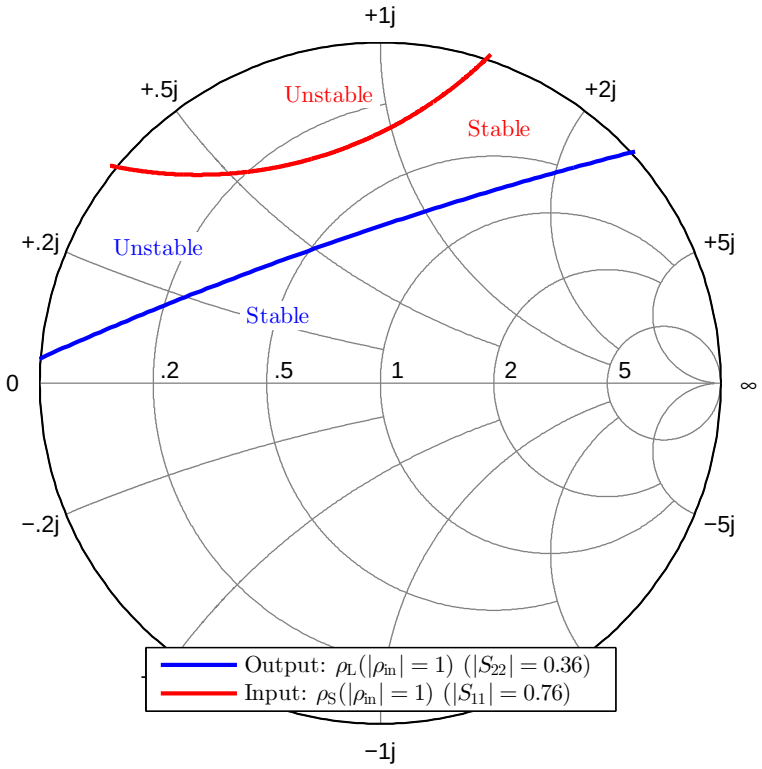

Figure 2: Input and output stability circles at $f = 2.5$ GHz.

Table 3: Resistor values needed to achieve unconditional stability and wanted characteristics at $f = 2.5$ GHz.

Port	Node	Connection	$R_{\text{uncond. stable}}$ [Ω]	$R_{\text{spec. met}}$ [Ω]
Input	Gate	Series	≥ 15.3	Too noisy
Input	Gate	Shunt	≤ 80.9	Too noisy
Output	Drain	Series	≥ 37.8	$69.8 \ldots 109$
Output	Drain	Shunt	≤ 6.27	Not enough gain

2.2 Can you use the unilateral or bilateral scenario for the design of the transistor amplifier? Justify your selection.

The unilateral figure of merit and the gain error resulting from (stabilized) unilateral approach are shown in Figures 3 and 4, respectively. At 2.5 GHz, $|S_{12}| = 0.0424$, $U = 0.0967$ and $-0.802 < G_T/G_{TU} < +0.884$ [dB]. Due to rather strict design goals, unilateral approach is not appropriate. For this reason, bilateral analysis was used. One cannot "go wrong" with bilateral analysis since it's valid even for truly unilateral designs.

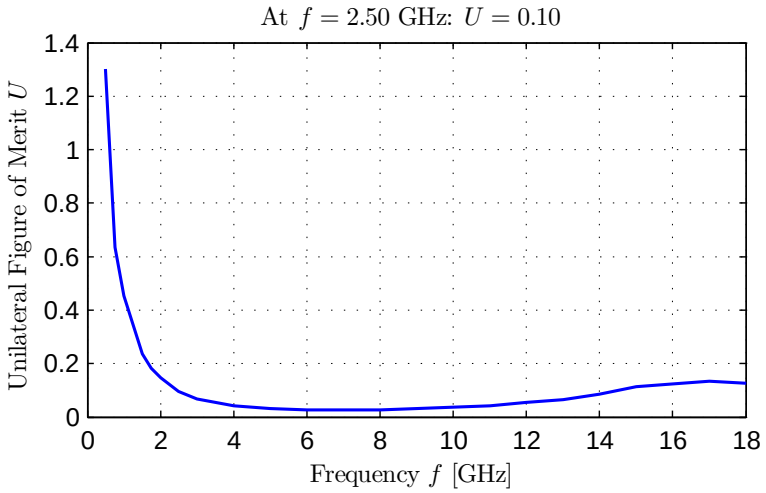

Figure 3: The unilateral figure of merit U as a function of frequency f.

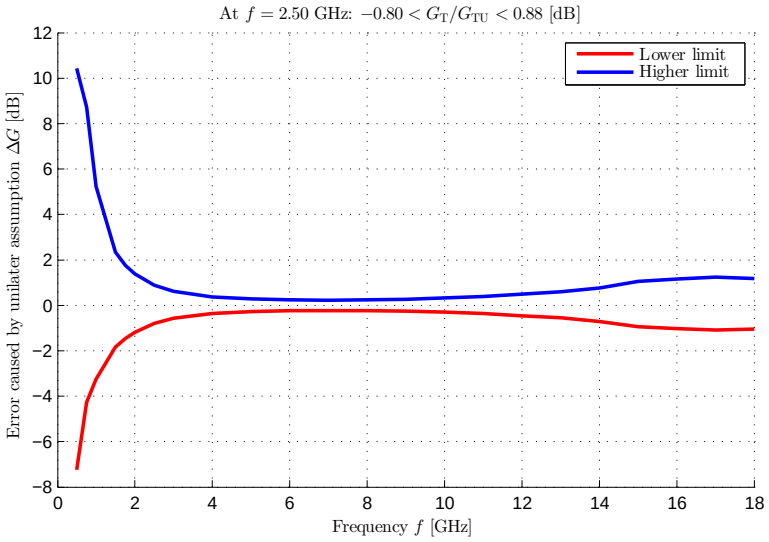

Figure 4: asd

2.3 Draw the gain and noise circles. Calculate the input and output reflection coefficients and design the matching networks. Does the input reflection coefficient satisfy the requirement? If not, what can be done?

The constant available gain and noise figure circles of the stabilized transistor are shown on the ρ_S Smith chart in Fig. 5, drawn using Eq. (12-17). Smaller circles are for larger G_As and smaller Fs. To meet gain and noise figure specifications, the source reflection coefficient

must be chosen from the area enclosed within both $G_\mathrm{A} = 13$ dB and $F = 0.8$ dB circles (both dashed). One may think of it as a Venn diagram of sorts.

$$C_\mathrm{A} = \frac{g_\mathrm{A} C_1^*}{1 + g_\mathrm{A}(|S_{11}|^2 - |\Delta|^2)} \quad \text{(center)} \tag{12}$$

$$R_\mathrm{A} = \frac{\sqrt{1 - 2K|S_{12}S_{21}|g_\mathrm{A} + |S_{12}S_{21}|^2 g_\mathrm{A}^2}}{|1 + g_\mathrm{A}(|S_{11}|^2 - |\Delta|^2|)} \quad \text{(radius)} \tag{13}$$

$$C_1 = S_{11} - \Delta S_{22}^* \quad \text{(used for } C_\mathrm{A}\text{)} \tag{14}$$

$$g_\mathrm{A} = \frac{G_\mathrm{A}}{|S_{21}|^2} \quad \text{(used for } C_\mathrm{A} \text{ and } R_\mathrm{A}\text{)} \tag{15}$$

$$C_F = \frac{\rho_\mathrm{opt}}{1 + N} \quad \text{(center)} \tag{16}$$

$$R_F = \frac{\sqrt{N^2 + N(1 - |\rho_\mathrm{opt}|)}}{1 + N} \quad \text{(radius)} \tag{17}$$

$$N = \frac{F - F_\mathrm{min}}{4R_\mathrm{n}/Z_0}|1 + \rho_\mathrm{opt}|^2 \quad \text{(used for } C_F \text{ and } R_F\text{)} \tag{18}$$

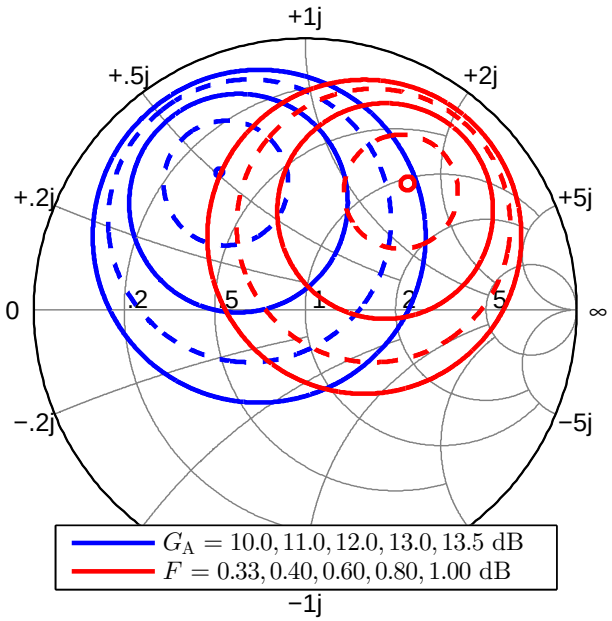

Figure 5: Constant available gain G_A and noise figure F circles at $f = 2.5$ GHz.

To meet the return loss specification, ρ_S must be chose quite close to gain maximum. In the pre-design, ρ_S was chosen with this in mind while the output was conjugate-matched. Chosen reflection coefficients and the properties for the pre-design are shown if Eq. (19-22) and Table

4, respectively. Formulae used to find RL_{in}, RL_{out} and F are shown in Eq. (23 - 25).

$$\rho_{\text{S}} = -0.191 + \text{j}0.461 = 0.499\angle{+112.6}^\circ \quad (19)$$
$$\rho_{\text{in}} = -0.307 - \text{j}0.503 = 0.589\angle{-121.5}^\circ \quad (20)$$
$$\rho_{\text{L}} = +0.461 + \text{j}0.171 = 0.492\angle{+20.4}^\circ \quad (21)$$
$$\rho_{\text{out}} = +0.461 - \text{j}0.171 = 0.492\angle{-20.4}^\circ \quad (22)$$

Table 4: Properties of the pre-design.

Parameter	Value [dB]	Design goal [dB]	Margin [dB]
RL_{in}	15.22	≥ 15	0.22
RL_{out}	∞	≥ 15	∞
G_{A}	13.35	≥ 13	0.35
G_{T}	13.35	≥ 13	0.35
G_{P}	13.49	≥ 13	0.49
F	0.737	≤ 0.8	0.062

$$RL_{\text{im}} = -10\lg\left(1 - \frac{P_{\text{in}}}{P_{\text{avs}}}\right)\ \text{dB} = -10\lg\left(1 - \frac{G_{\text{T}}}{G_{\text{P}}}\right)\ \text{dB} \quad (23)$$
$$RL_{\text{out}} = -10\lg\left(1 - \frac{P_{\text{L}}}{P_{\text{avn}}}\right)\ \text{dB} = -10\lg\left(1 - \frac{G_{\text{T}}}{G_{\text{A}}}\right)\ \text{dB} \quad (24)$$
$$F = 10\lg\left(F_{\text{min}} + \frac{4R_{\text{n}}}{Z_0}\frac{|\rho_{\text{S}} - \rho_{\text{opt}}|^2}{(1 - |\rho_{\text{S}}|^2)|1 + \rho_{\text{opt}}|^2}\right)\ \text{dB} \quad (25)$$

Thus it is in theory possible to achieve all design goals, but only when the transistor is stabilized with a suitable series resistor connected to the drain. Increasing this resistance moves constant gain and noise figure circles close together while decreasing total gain. Without this resistor the circles are far apart, and the specifications cannot be met. Tuning the value of this transistor one may optimize the design for desired characteristics. A gain high as 14.7 dB and a noise figure as low as 0.722 dB may be obtained separately, when zero-margins are accepted for all other possible criteria.

If this stabilizing resistor cannot be used, one can always resort to balanced amplifiers and IDCS designs, but they both have their own drawbacks. In practice the situation will be more difficult as any practical realization of DC-biasing will affect the performance characteristics.

For simplicity, open parallel stub matching is used to realize both input and output matching circuits. Traditional matching process (where one moves toward the generator) may be used, if we match the complex conjugates instead the actual values. Thus, the impedances to be matched are

$$\rho_{\text{S}}^* = -0.191 - \text{j}0.461 = 0.499\angle{-112.6}^\circ \Rightarrow z_{\text{S}}^* = 0.460 - \text{j}0.565, \quad (26)$$
$$\rho_{\text{L}}^* = +0.461 - \text{j}0.171 = 0.492\angle{-20.4}^\circ \Rightarrow z_{\text{L}}^* = 2.371 - \text{j}1.070. \quad (27)$$

The matching process is shown on the Smith charts in Figures 6 and 7. The stub distances/lengths are also shown in Table 5.

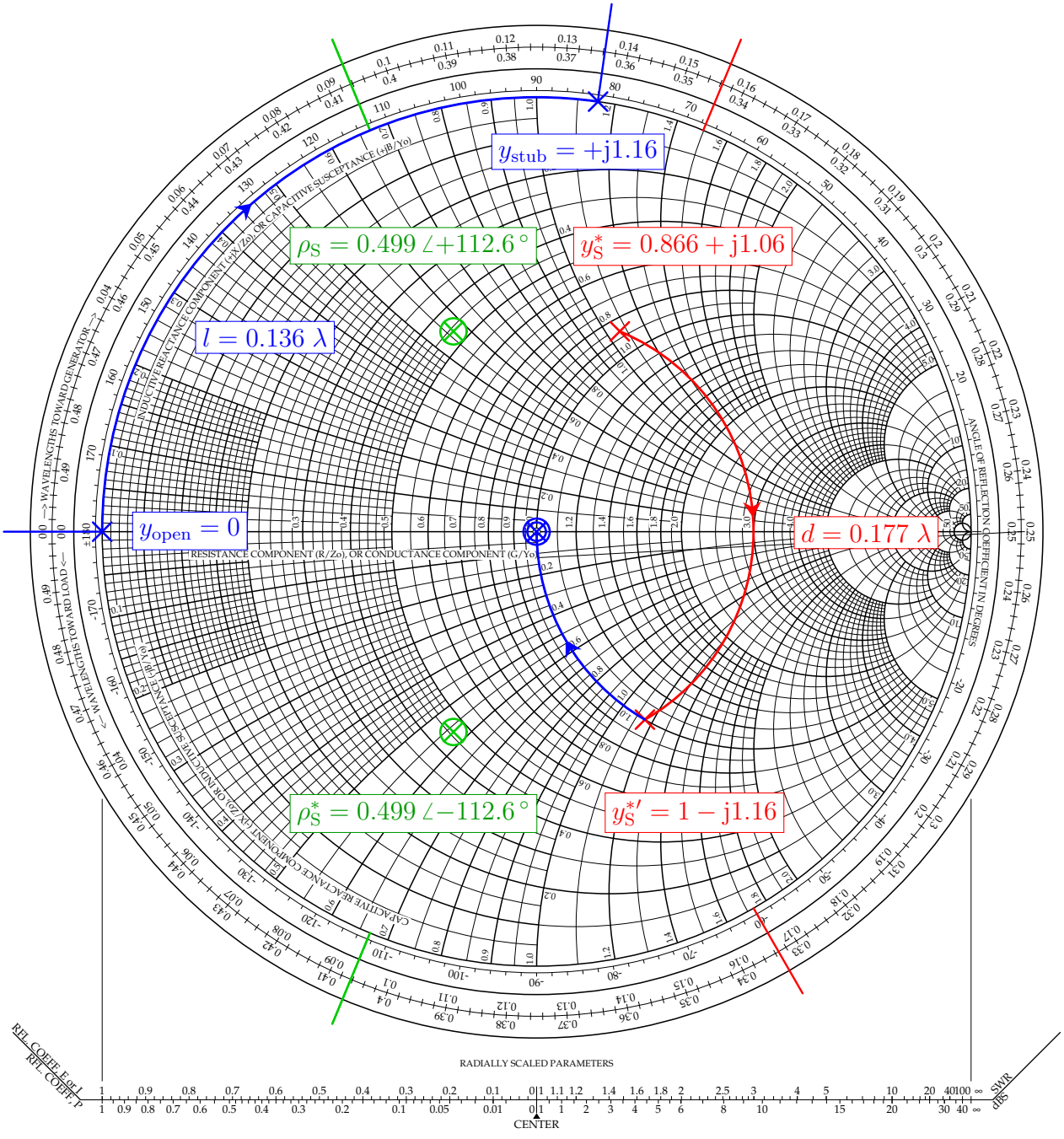

Figure 6: Input matching circuit using open-circuited parallel stubs.

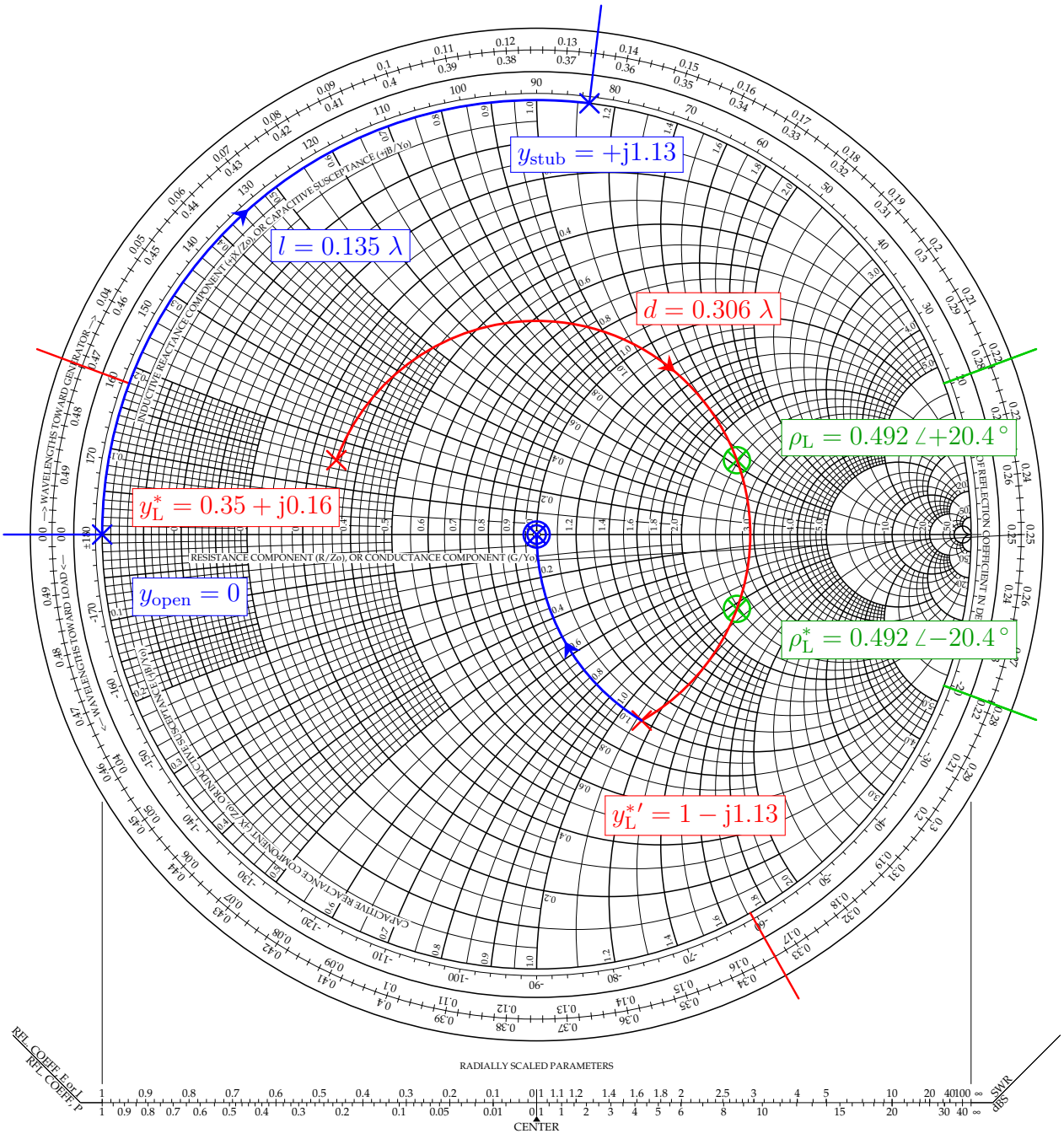

Figure 7: Output matching circuit using open-circuited parallel stubs.

Table 5: Matching circuits of the pre-design ($Z_0 = 50\ \Omega$).

Port	Distance from transistor d [λ]	Stub length l [λ]
Input	0.177	0.136
Output	0.306	0.135

2.4 Draw the final schematic of the amplifier including the matching circuits.

A circuit schematic of the rough pre-design, including stabilization resistor and matching circuits, is shown in Fig. 8.

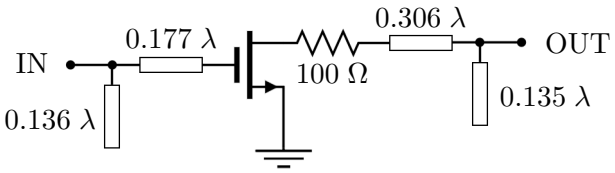

Figure 8: Amplifier pre-design ($Z_0 = 50\ \Omega$).

References

[1] I. J. Bahl, *Fundamentals of RF and Microwave Transistor Amplifiers*, J. Wiley & Sons, 2009.

[2] G. Gonzalez, *Microwave Transistor Amplifiers – Analysis and Design*, Prentice Hall, 2nd Ed., 1997.

[3] D. M. Pozar, *Microwave Engineering*, J. Wiley & Sons, 4th Ed., 2012.

Appendix A: MATLAB source code

The source code MATLAB script used in the pre-design process is shown below. Apart from `exportfig`, all other features are self-made. No RF toolbox is required, meaning the script might also be compatible with GNU Octave.

```
% RF Lab 4: Amplifier - Pre-study
% Tuomas Leinonen
% 84695P

% As the indexing/automation got rather complicated at times,
% the code contains some "manual-tuning".

clear;
clc;
%close all;

% Save figures
figs = struct( ...
    'global',   false, ...
    'mu',       true, ...
    'SLstab',   true, ...
    'Sstab',    false, ...
    'Lstab',    false, ...
    'UFM',      true, ...
    'DeltaG',   true, ...
    'MSG',      false, ...
    'GFcirc',   true, ...
    'GPcirc',   false ...
    );

% Design specifications
dParam = struct( ...
    'f',        2.5e9, ...
    'Vds',      2, ...
    'Ids',      30e-3, ...
    'Gmin',     13, ...
    'Fmax',     0.8, ...
    'RLmin',    15, ...
    'h',        0.787e-3, ...
    'er',       2.3, ...
    'stab',     true ...
    );

% Paramaters of the original design
orig = struct( ...
    'S',        [], ...
    'ABCD',     [], ...
    'Delta',    [], ...
    'K',        [], ...
    'mu',       [], ...
```

```
    'CL',        [], ...
    'RL',        [], ...
    'CS',        [], ...
    'RS',        [] ...
    );

% Paramaters of the stabilized design
stab = struct( ...
    'S',         [], ...
    'ABCD',      [], ...
    'Delta',     [], ...
    'K',         [], ...
    'mu',        [], ...
    'CL',        [], ...
    'RL',        [], ...
    'CS',        [], ...
    'RS',        [] ...
    );

interpFunc = 'pchip';   % Interpolation function

% S: S11 S21 S12 S22
[f1, S, Z0] = readSparamFile('2V_30mA_S.s2p');
[f2, Fm, Ro, Rn] = readNoiseParamFile('2V_30mA_F.s2p');
orig.S = S;

% Transmission parameters
ABCD = repmat(1 ./ (2 * S(:, 2)), 1, 4) .* [ ...
    (1 + S(:, 1)) .* (1 - S(:, 4)) + S(:, 2) .* S(:, 3), ...
    Z0 * ( (1 + S(:, 1)) .* (1 + S(:, 4)) - S(:, 2) .* S(:, 3) ), ...
    1/Z0 * ( (1 - S(:, 1)) .* (1 - S(:, 4)) - S(:, 2) .* S(:, 3) ), ...
    (1 - S(:, 1)) .* (1 + S(:, 4)) + S(:, 2) .* S(:, 3) ...
    ];
orig.ABCD = ABCD;

% Stability
Delta = S(:, 1) .* S(:, 4) - S(:, 2) .* S(:, 3);
K = (1 - abs(S(:, 1)).^2 - abs(S(:, 4)).^2 + abs(Delta).^2) ./ ( 2 *
   abs(S(:, 2) .* S(:, 3)) );
mu = (1 - abs(S(:, 1)).^2) ./ ( abs(S(:, 4) - Delta .* conj(S(:, 1))) +
   abs(S(:, 2) .* S(:, 3)) );
orig.Delta = Delta;
orig.K = K;
orig.mu = mu;

Rollet = ((abs(Delta) < 1) & (K > 1));
UST = (mu > 1);
Fstab = f1(Rollet == 1);
Sstr = {'Potentially unstable', 'Unconditionally stable'};

figure(1);
```

```
clf;
plot(f1 / 1e9, mu, 'color', 'b', 'linewidth', 2);
grid on;
ylabel('Stability $$\mu$$', 'interpreter', 'latex');
xlabel('Frequency $$f$$ [GHz]', 'interpreter', 'latex');
title(sprintf('At $$f = %.2f$$ GHz: $$\\mu = %.2f$$, $$K = %.2f$$ and
   $$|\\Delta| = %.2f$$ $$\\Rightarrow$$ %s', ...
    dParam.f/1e9 , mu(f1 == dParam.f), K(f1 == dParam.f), abs(Delta(f1 ==
       dParam.f)), Sstr{1 + UST(f1 == dParam.f)}), 'interpreter', 'latex');

% Save figure
if figs.global && figs.mu
exportfig(gcf, 'mu.eps', ...
    'width',        12, ...
    'height',       6, ...
    'color',        'rgb', ...
    'resolution',   300, ...
    'LockAxes',     1, ...
    'FontMode',     'fixed', ...
    'FontSize',     9, ...
    'LineMode',     'scaled'    );
end

% Stability circles
CL = conj( S(:, 4) - Delta .* conj(S(:, 1)) ) ./ ( abs(S(:, 4)).^2 -
   abs(Delta).^2 );
RL = abs( S(:, 2) .* S(:, 3) ./ ( abs(S(:, 4)).^2 - abs(Delta).^2 ) );
CS = conj( S(:, 1) - Delta .* conj(S(:, 4)) ) ./ ( abs(S(:, 1)).^2 -
   abs(Delta).^2 );
RS = abs( S(:, 2) .* S(:, 3) ./ ( abs(S(:, 1)).^2 - abs(Delta).^2 ) );

orig.CL = CL;
orig.RL = RL;
orig.CS = CS;
orig.RS = RS;

figure(2);
clf;
initSmithChart(gca);
hold on;
h = [0 0];

C = CL(f1 == dParam.f);
R = RL(f1 == dParam.f);
p = angle(C) + linspace(0, 2 * pi, 4001);
p = p(1 : end-1);
z = C + R * exp(1i * p);
z = z(abs(z) <= 1.02);
if ~isempty(z)
    h(1) = plot(gca, real(z), imag(z), ...
        'color',        'b', ...
```

```
        'linewidth',    2);
end

C = CS(f1 == dParam.f);
R = RS(f1 == dParam.f);
p = angle(C) + linspace(0, 2 * pi, 4001);
p = p(1 : end-1);
z = C + R * exp(1i * p);
z = z(abs(z) <= 1.02);
if ~isempty(z)
    h(2) = plot(gca, real(z), imag(z), ...
        'color',        'r', ...
        'linewidth',    2);
end
hold off;

text(0.35, 0.75, 'Stable', ...
    'HorizontalAlignment', 'center', ...
    'color',    'r', ...
    'backgroundColor',  'white', ...
    'interpreter', 'latex');
text(-0.15, 0.85, 'Unstable', ...
    'HorizontalAlignment', 'center', ...
    'color',    'r', ...
    'backgroundColor',  'white', ...
    'interpreter', 'latex');
text(-0.3, 0.2, 'Stable', ...
    'HorizontalAlignment', 'center', ...
    'color',    'b', ...
    'backgroundColor',  'white', ...
    'interpreter', 'latex');
text(-0.65, 0.4, 'Unstable', ...
    'HorizontalAlignment', 'center', ...
    'color',    'b', ...
    'backgroundColor',  'white', ...
    'interpreter', 'latex');

legend(gca, h, ...
    {sprintf('Output: $$\\rho_\\mathrm{L}(|\\rho_\\mathrm{in}| = 1$$)
       ($$|S_{22}| = %.2f$$)', abs(S(f1 == dParam.f, 4))), ...
    sprintf('Input: $$\\rho_\\mathrm{S}(|\\rho_\\mathrm{in}| = 1$$)
       ($$|S_{11}| = %.2f$$)', abs(S(f1 == dParam.f, 1)))}, ...
    'interpreter',  'latex', ...
    'location',     'south');

xlim([-1 1]);
ylim([-1 1]);

if figs.global && figs.SLstab
exportfig(gcf, 'SLstab.eps', ...
    'width',        12, ...
```

```
    'height',        12, ...
    'color',         'rgb', ...
    'resolution',    300, ...
    'LockAxes',      1, ...
    'FontMode',      'fixed', ...
    'FontSize',      9, ...
    'LineMode',      'scaled'    );
end


figure(3);
clf;
initSmithChart(gca);
hold on;

for i = 1 : length(f1)
    lineStyle = repmat('-', 1, 2 - mod(i, 2));
    C = CL(i);
    R = RL(i);
    p = angle(C) + linspace(0, 2 * pi, 4001);
    p = p(1 : end-1);
    z = C + R * exp(1i * p);
    z = z(abs(z) <= 1.02);
    if ~isempty(z)
    plot(gca, real(z), imag(z), ...
        'color',         'b', ...
        'lineStyle',     lineStyle, ...
        'linewidth',     2);
    end
end
hold off;
xlim([-1 1]);
ylim([-1 1]);
text(0, -0.2, '\quad Output stability circles: $$|S_{22}| < 1$$ \quad', ...
    'HorizontalAlignment', 'center', ...
    'backgroundColor',  'white', ...
    'interpreter', 'latex');

if figs.global && figs.Lstab
exportfig(gcf, 'Lstab.eps', ...
    'width',         12, ...
    'height',        12, ...
    'color',         'rgb', ...
    'resolution',    300, ...
    'LockAxes',      1, ...
    'FontMode',      'fixed', ...
    'FontSize',      9, ...
    'LineMode',      'scaled'    );
end

figure(4);
```

```
clf;
initSmithChart(gca);
hold on;

for i = 1 : length(f1)
    lineStyle = repmat('-', 1, 2 - mod(i, 2));
    C = CS(i);
    R = RS(i);
    p = angle(C) + linspace(0, 2 * pi, 4001);
    p = p(1 : end-1);
    z = C + R * exp(1i * p);
    z = z(abs(z) <= 1.02);
    plot(gca, real(z), imag(z), ...
        'color',        'r', ...
        'lineStyle',    lineStyle, ...
        'linewidth',    2);
end
hold off;
xlim([-1 1]);
ylim([-1 1]);
text(0, -0.2, '\quad Input stability circles: $$|S_{11}| < 1$$ \quad', ...
    'HorizontalAlignment', 'center', ...
    'backgroundColor',  'white', ...
    'interpreter', 'latex');

if figs.global && figs.Sstab
exportfig(gcf, 'Sstab.eps', ...
    'width',        18, ...
    'height',       18, ...
    'color',        'rgb', ...
    'resolution',   300, ...
    'LockAxes',     1, ...
    'FontMode',     'fixed', ...
    'FontSize',     9, ...
    'LineMode',     'scaled'    );
end

% Stabilization

% Stabilization resistors
% SerRS >= 15.3, ParRS <= 80.9, SerRL >= 37.8, ParRL <= 6.27
% To meet specifications: SerRL = [69.8, 109]
SerRS = [   15.3,   0,      0,      0       ];
ParRS = [   Inf,    80.9,   Inf,    Inf     ];
SerRL = [   0,      0,      100,    0       ];
ParRL = [   Inf,    Inf,    Inf,    6.27    ];

for j = 3 %1 : 4
    ABCDs = [1,     SerRS(j);   1/ParRS(j),     1];
    ABCDl = [1,     SerRL(j);   1/ParRL(j),     1];

```

```
293     ABCD2 = zeros(size(ABCD));
294     %KK = [0 0; 0 0];
295
296     for i = 1 : length(f1)
297         KK = ABCDs * [ABCD(i, 1), ABCD(i, 2); ABCD(i, 3), ABCD(i, 4)] *
                ABCDl;
298         ABCD2(i,:) = [KK(1, 1), KK(1, 2), KK(2, 1), KK(2, 2)];
299     end
300
301     % Revert back to S-param
302     SS = repmat(1 ./ ( ABCD2(:, 1) + ABCD2(:, 2) / Z0 + ABCD2(:, 3) * Z0 +
            ABCD2(:, 4) ), 1, 4) .* [ ...
303         ABCD2(:, 1) + ABCD2(:, 2) / Z0 - ABCD2(:, 3) * Z0 - ABCD2(:, 4), ...
304         repmat(2, size(ABCD, 1), 1), ...
305         2 * (ABCD2(:, 1) .* ABCD2(:, 4) - ABCD2(:, 2) .* ABCD2(:, 3)), ...
306         -ABCD2(:, 1) + ABCD2(:, 2) / Z0 - ABCD2(:, 3) * Z0 + ABCD2(:, 4) ...
307         ];
308
309     Delta2 = SS(:, 1) .* SS(:, 4) - SS(:, 2) .* SS(:, 3);
310     K2 = (1 - abs(SS(:, 1)).^2 - abs(SS(:, 4)).^2 + abs(Delta2).^2) ./ ( 2
            * abs(SS(:, 2) .* SS(:, 3)) );
311     mu2 = (1 - abs(SS(:, 1)).^2) ./ ( abs(SS(:, 4) - Delta2 .* conj(SS(:,
            1))) + abs(SS(:, 2) .* SS(:, 3)) );
312
313     fprintf('j = %d: mu = %.6f, G0 = %.2f dB\n', j, mu2(f1 == dParam.f),
            20*log10(SS(f1 == dParam.f, 2)));
314 end
315
316 % Use stabilized or original design
317 if dParam.stab
318     S = SS;
319     Delta = Delta2;
320     K = K2;
321     mu = mu2;
322     CL = conj( S(:, 4) - Delta .* conj(S(:, 1)) ) ./ ( abs(S(:, 4)).^2 -
            abs(Delta).^2 );
323     RL = abs( S(:, 2) .* S(:, 3) ./ ( abs(S(:, 4)).^2 - abs(Delta).^2 ) );
324     CS = conj( S(:, 1) - Delta .* conj(S(:, 4)) ) ./ ( abs(S(:, 1)).^2 -
            abs(Delta).^2 );
325     RS = abs( S(:, 2) .* S(:, 3) ./ ( abs(S(:, 1)).^2 - abs(Delta).^2 ) );
326
327     stab.S = SS;
328     stab.Delta = Delta2;
329     stab.K = K2;
330     stab.mu = mu2;
331     stab.CL = CL;
332     stab.RL = RL;
333     stab.CS = CS;
334     stab.RS = CS;
335 end
336
```

```
% Unilateral figure of merit and the error caused by unilater approach
U = abs( S(:, 1) .* S(:, 2) .* S(:, 3) .* S(:, 4) ) ./ ...
    ( (1 - abs(S(:, 1)).^2) .* (1 - abs(S(:, 4)).^2) );
DiffLow = 1 ./ (1 + U).^2;
DiffHigh = 1 ./ (1 - U).^2;

figure(5);
clf;
plot(f1 / 1e9, U,    ...
    'color',         'b', ...
    'linewidth',     2);
grid on;
ylabel('Unilateral Figure of Merit $$U$$', 'interpreter', 'latex');
xlabel('Frequency $$f$$ [GHz]', 'interpreter', 'latex');
title(sprintf('At $$f = %.2f$$ GHz: $$U = %.2f$$', ...
    dParam.f/1e9 , U(f1 == dParam.f)), 'interpreter', 'latex');

if figs.global && figs.UFM
exportfig(gcf, 'UFM.eps', ...
    'width',         12, ...
    'height',        6, ...
    'color',         'rgb', ...
    'resolution',    300, ...
    'LockAxes',      1, ...
    'FontMode',      'fixed', ...
    'FontSize',      9, ...
    'LineMode',      'scaled'    );
end


figure(6);
clf;
hold on;
plot(f1 / 1e9, 10*log10(DiffLow),    ...
    'color',         'r', ...
    'linewidth',     2);
plot(f1 / 1e9, 10*log10(DiffHigh),   ...
    'color',         'b', ...
    'linewidth',     2);
hold off;
grid on;
ylabel('Error caused by unilater assumption $$\Delta G$$ [dB]',
   'interpreter', 'latex');
xlabel('Frequency $$f$$ [GHz]', 'interpreter', 'latex');
title(sprintf('At $$f = %.2f$$ GHz: $$%.2f < G_\\mathrm{T}/G_\\mathrm{TU} <
   %.2f$$ [dB]', ...
    dParam.f/1e9 , 10*log10(DiffLow(f1 == dParam.f)), 10*log10(DiffHigh(f1
       == dParam.f))), 'interpreter', 'latex');
xlim([0 18]);
%set(gca, 'XTick', 0:2:18);

```

```
385
386 legend(gca, ...
387     {'Lower limit', 'Higher limit'}, ...
388     'interpreter',  'latex', ...
389     'location',     'northeast');
390
391 if figs.global && figs.DeltaG
392 exportfig(gcf, 'DeltaG.eps', ...
393     'width',        18, ...
394     'height',       10, ...
395     'color',        'rgb', ...
396     'resolution',   300, ...
397     'LockAxes',     1, ...
398     'FontMode',     'fixed', ...
399     'FontSize',     9, ...
400     'LineMode',     'scaled'    );
401 end
402
403 % MSG
404 MSG = abs(S(:, 2) ./ S(:, 3));
405
406 figure(7);
407 clf;
408 plot(f1 / 1e9, 10*log10(MSG), ...
409     'color',        'b', ...
410     'linewidth',    2);
411 grid on;
412 ylabel('Maximum Stable Gain $$\mathit{MSG}$$ [dB]', 'interpreter', 'latex');
413 xlabel('Frequency $$f$$ [GHz]', 'interpreter', 'latex');
414 title(sprintf('At $$f = %.2f$$ GHz: $$\\mathit{MSG} = %.1f$$ dB', ...
415     dParam.f/1e9, 10*log10(MSG(f1 == dParam.f))), 'interpreter', 'latex');
416
417 if figs.global && figs.MSG
418 exportfig(gcf, 'MSG.eps', ...
419     'width',        12, ...
420     'height',       6, ...
421     'color',        'rgb', ...
422     'resolution',   300, ...
423     'LockAxes',     1, ...
424     'FontMode',     'fixed', ...
425     'FontSize',     9, ...
426     'LineMode',     'scaled'    );
427 end
428
429 % Constant Available Gain & Noise Figure Circles
430 % (Indexing order not "ideal")
431 figure(8);
432 clf;
433 initSmithChart(gca);
434 hold on;
435 h = [0 0 0];
```

```
436
437 C = CS(f1 == dParam.f);
438 R = RS(f1 == dParam.f);
439 p = angle(C) + linspace(0, 2 * pi, 4001);
440 p = p(1 : end-1);
441 z = C + R * exp(1i * p);
442 z = z(abs(z) <= 1.02);
443 if ~isempty(z)
444     h(1) = plot(gca, real(z), imag(z), ...
445         'color',        'k', ...
446         'linewidth',    2);
447 end
448
449 %vGA = [12 13 14 15 16 17 18];
450 vGA = [10 11 12 13 13.5];
451 vFi = [0.331 0.4 0.6 0.8 1];
452
453 vGA = 10 .^ (vGA / 10);
454 vFi = 10 .^ (vFi / 10);
455
456 C1 = S(:, 1) - Delta .* conj(S(:, 4));
457
458 hh = zeros(size(vGA));
459 for i = 1 : length(vGA)
460     GA = vGA(i);
461     lineStyle = repmat('-', 1, 2 - mod(i, 2));
462     %ga = ( 1 - abs(Rs).^2 ) ./ ( 1 - S(:, 4).^2 + abs(Rs).^2 .* ( abs(S(:,
        1).^2 - abs(Delta).^2) - 2 * real(Rs .* C1) );
463     ga = GA ./ abs(S(:, 2)).^2;
464
465     Ca = ( ga .* conj(C1) ) ./ ( 1 + ga .* ( abs(S(:, 1)).^2 -
        abs(Delta).^2 ) );
466     Ra = sqrt( 1 - 2 * K .* abs( S(:, 2) .* S(:, 3) ) .* ga + abs( S(:, 2)
        .* S(:, 3) ).^2 .* ga.^2 ) ./ ...
467         abs( 1 + ga .* (abs(S(:, 1)).^2 - abs(Delta).^2 ) );
468
469     C = Ca(f1 == dParam.f);
470     R = abs(Ra(f1 == dParam.f));
471     p = angle(C) + linspace(0, 2 * pi, 4001);
472     p = p(1 : end-1);
473     z = C + R * exp(1i * p);
474     z = z(abs(z) <= 1.02);
475     if ~isempty(z)
476         hh(i) = plot(gca, real(z), imag(z), ...
477             'color',        'b', ...
478             'lineStyle',    lineStyle, ...
479             'linewidth',    2);
480     end
481 end
482 h(2) = hh(1);
483
```

```
484 hh = zeros(size(vFi));
485 for i = 1 : length(vFi)
486     Fi = vFi(i);
487     lineStyle = repmat('-', 1, 2 - mod(i, 2));
488     Ni = (Fi - Fm) ./ (4 * Rn) .* abs(1 + Ro).^2;
489     CFi = Ro ./ (1 + Ni);
490     RFi = sqrt(Ni.^2 + Ni .* (1 - abs(Ro).^2)) ./ (1 + Ni);
491
492     C = CFi(f1 == dParam.f);
493     R = abs(RFi(f1 == dParam.f));
494     p = angle(C) + linspace(0, 2 * pi, 4001);
495     p = p(1 : end-1);
496     z = C + R * exp(1i * p);
497     z = z(abs(z) <= 1.02);
498     if ~isempty(z)
499         hh(i) = plot(gca, real(z), imag(z), ...
500             'color',        'r', ...
501             'lineStyle',    lineStyle, ...
502             'linewidth',    2);
503     end
504 end
505 hold off;
506 h(3) = hh(1);
507 if mu(f1 == dParam.f) > 1
508     h = h(2:3);
509 end;
510
511 Fstr = sprintf(', %.2f', 10*log10(vFi));
512 Gstr = sprintf(', %.1f', 10*log10(vGA));
513
514 legend(gca, h, ...
515     {sprintf('$$G_\\mathrm{A} = %s$$ dB \\quad', Gstr(3:end)), ...
516     sprintf('$$F = %s$$ dB \\qquad', Fstr(3:end)), ...
517     sprintf('Input stability circle: $$|S_{11}| = %.2f$$ \\quad', abs(S(f1
        == dParam.f, 1)))}, ...
518     'interpreter',  'latex', ...
519     'location',     'south');
520
521 if figs.global && figs.GFcirc
522 exportfig(gcf, 'GFcirc.eps', ...
523     'width',        12, ...
524     'height',       12, ...
525     'color',        'rgb', ...
526     'resolution',   300, ...
527     'LockAxes',     1, ...
528     'FontMode',     'fixed', ...
529     'FontSize',     9, ...
530     'LineMode',     'scaled'    );
531 end
532
533
```

```
% Constant Operating Gain Circles
figure(9);
clf;
initSmithChart(gca);
hold on;
h = [0 0];

C = CL(f1 == dParam.f);
R = RL(f1 == dParam.f);
p = angle(C) + linspace(0, 2 * pi, 4001);
p = p(1 : end-1);
z = C + R * exp(1i * p);
z = z(abs(z) <= 1.02);
if ~isempty(z)
    h(1) = plot(gca, real(z), imag(z), ...
        'color',        'k', ...
        'linewidth',    2);
end

%vGP = [12 13 14 15 16 17 18];
vGP = [10 11 12 13 13.5];
vGP = 10 .^ (vGP / 10);

C2 = S(:, 4) - Delta .* conj(S(:, 1));

hh = zeros(size(vGP));
for i = 1 : length(vGP)
    GP = vGP(i);
    lineStyle = repmat('-', 1, 2 - mod(i, 2));
    %ga = ( 1 - abs(Rs).^2 ) ./ ( 1 - S(:, 4).^2 + abs(Rs).^2 .* ( abs(S(:,
       1).^2 - abs(Delta).^2) - 2 * real(Rs .* C1) );
    gp = GP ./ abs(S(:, 2)).^2;

    Cp = ( gp .* conj(C2) ) ./ ( 1 + gp .* ( abs(S(:, 4)).^2 -
       abs(Delta).^2 ) );
    Rp = sqrt( 1 - 2 * K .* abs( S(:, 2) .* S(:, 3) ) .* gp + abs( S(:, 2)
       .* S(:, 3) ).^2 .* gp.^2 ) ./ ...
        abs( 1 + gp .* (abs(S(:, 4)).^2 - abs(Delta).^2 ) );

    C = Cp(f1 == dParam.f);
    R = Rp(f1 == dParam.f);
    p = angle(C) + linspace(0, 2 * pi, 4001);
    p = p(1 : end-1);
    z = C + R * exp(1i * p);
    z = z(abs(z) <= 1.02);
    if ~isempty(z)
        hh(i) = plot(gca, real(z), imag(z), ...
            'color',        'b', ...
            'lineStyle',    lineStyle, ...
            'linewidth',    2);
    end
```

```
end
h(2) = hh(1);
if mu(f1 == dParam.f) > 1
    h = h(2);
end;

Gstr = sprintf(', %.1f', 10*log10(vGP));

legend(gca, h, ...
    {sprintf('$$G_\\mathrm{P} = %s$$ dB \\quad', Gstr(3:end)), ...
    sprintf('Output stability circle: $$|S_{22}| = %.2f$$ \\qquad',
       abs(S(f1 == dParam.f, 4)))}, ...
    'interpreter',  'latex', ...
    'location',     'south');

if figs.global && figs.GPcirc
exportfig(gcf, 'GFcirc.eps', ...
    'width',        12, ...
    'height',       12, ...
    'color',        'rgb', ...
    'resolution',   300, ...
    'LockAxes',     1, ...
    'FontMode',     'fixed', ...
    'FontSize',     9, ...
    'LineMode',     'scaled'    );
end

% Constant VSWR circles -> Needs Rin and Rout
% vRL = [10 15 20];
% Rab = 10^(-vRL/20);

% CVi = conj(Rin) .* (1 - Rab.^2) ./ (1 - abs(Rab .* Rin));
% RVi = Rab .* (1 - abs(Rin).^2) ./ (1 - abs(Rab .* Rin));
% CVo = conj(Rout) .* (1 - Rab.^2) ./ (1 - abs(Rab .* Rout));
% RVo = Rab .* (1 - abs(Rout).^2) ./ (1 - abs(Rab .* Rout));


% Gains & Noise figure for chosen Rs and Rl
% Is it stable inside or outside the stability circle?
sInOut = ((abs(S(:, 1)) < 1) & abs(CS) < RS) | ((abs(S(:, 1)) > 1) &
   abs(CS) > RS);
lInOut = ((abs(S(:, 4)) < 1) & abs(CL) < RL) | ((abs(S(:, 4)) > 1) &
   abs(CL) > RL);
margin = 0.05;
YesNo = {'Yes', 'No'};

% Find Rs for certain F or G_A
% Would be better if different values for Rs were chosen from a
% line connecting G_A,max and F_min points
Fi = 10^(0.737/10);
Ni = (Fi - Fm) ./ (4 * Rn) .* abs(1 + Ro).^2;
```

```
630 CFi = Ro ./ (1 + Ni);
631 RFi = sqrt(Ni.^2 + Ni .* (1 - abs(Ro).^2)) ./ (1 + Ni);
632 C = CFi(f1 == dParam.f);
633 R = abs(RFi(f1 == dParam.f));
634
635 % GA = 10^(14.7/10);
636 % ga = GA ./ abs(S(:, 2)).^2;
637 % Ca = ( ga .* conj(C1) ) ./ ( 1 + ga .* ( abs(S(:, 1)).^2 - abs(Delta).^2
      ) );
638 % Ra = sqrt( 1 - 2 * K .* abs( S(:, 2) .* S(:, 3) ) .* ga + abs( S(:, 2) .*
      S(:, 3) ).^2 .* ga.^2 ) ./ ...
639 %   abs( 1 + ga .* (abs(S(:, 1)).^2 - abs(Delta).^2 ) );
640 % C = Ca(f1 == dParam.f);
641 % R = abs(Ra(f1 == dParam.f));
642
643 p = angle(C) + linspace(0, 2 * pi, 3601);
644 p = p(1 : end-1);
645 z = C + R * exp(1i * p);
646 z = z(abs(z) < 1);
647 i = find(f1 == dParam.f);
648
649 for Rs = z
650     %Rs = 0.55 + 0.62 * 1i; %-0.3 + 0.8 * 1i; % Ro(f1 == dParam.f);
651     Rout = S(:, 4) + (S(:, 2) .* S(:, 3) .* Rs) ./ (1 - S(:, 1) .* Rs);
652     Rl = conj(Rout((f1 == dParam.f)));
653     Rin = S(:, 1) + (S(:, 2) .* S(:, 3) .* Rl) ./ (1 - S(:, 4) .* Rl);
654
655     Sstable = sqrt((real(CS) - real(Rs)).^2 + (imag(CS) - imag(Rs)).^2);
656     Lstable = sqrt((real(CL) - real(Rl)).^2 + (imag(CL) - imag(Rl)).^2);
657
658     Sstable = sInOut & (Sstable < (RS - margin)) | ~sInOut & (Sstable > (RS
        + margin));
659     Lstable = lInOut & (Lstable < (RL - margin)) | ~lInOut & (Lstable > (RL
        + margin));
660
661     Ga = abs(S(:, 2)).^2 .* (1 - abs(Rs).^2) ./ ( abs(1 - S(:, 1) .* Rs).^2
        .* (1 - abs(Rout).^2) );
662     Gt = abs(S(:, 2)).^2 .* (1 - abs(Rs).^2) .* (1 - abs(Rl).^2) ./ ...
663       ( abs(1 - Rs .* Rin).^2 .* abs(1 - S(:, 4) .* Rl).^2 );
664     Gp = abs(S(:, 2)).^2 .* (1 - abs(Rl).^2) ./ ...
665       ( (1 - abs(Rin).^2) .* abs(1 - S(:, 4) .* Rl).^2 );
666     GadB = 10*log10(Ga);
667     GtdB = 10*log10(Gt);
668     GpdB = 10*log10(Gp);
669
670     F = Fm + 4 * Rn .* abs(Rs - Ro).^2 ./ ( (1 - abs(Rs).^2) .* abs(1 +
        Ro).^2 );
671     FdB = 10*log10(F);
672
673     RLi = 1 ./ (1 - Gt ./ Gp);
674     RLo = 1 ./ (1 - Gt ./ Ga);
```

```
675     RLi(abs(RLi) > 1e12) = Inf; % For limited accuracy
676     RLo(abs(RLo) > 1e12) = Inf;
677     RLidB = 10 * log10(RLi);
678     RLodB = 10 * log10(RLo);
679
680     % Practical design
681     if (RLidB(f1 == dParam.f) > dParam.RLmin && ...
682             RLodB(f1 == dParam.f) > dParam.RLmin && ...
683             GtdB(f1 == dParam.f) > dParam.Gmin && ...
684             FdB(f1 == dParam.f) < dParam.Fmax) && ...
685             abs(Rs) < 1 && Sstable(f1 == dParam.f) && ...
686             abs(Rl) < 1 && Lstable(f1 == dParam.f) && ...
687             MSG(f1 == dParam.f) > Gt(f1 == dParam.f)
688
689         fprintf('Rs   = %+.3f %+.3fj (%.3f /_ %+.2f deg)\n', real(Rs),
               imag(Rs), abs(Rs), angle(Rs) / pi * 180);
690         fprintf('Rin  = %+.3f %+.3fj (%.3f /_ %+.2f deg)\n', real(Rin(f1 ==
               dParam.f)), imag(Rin(f1 == dParam.f)), abs(Rin(f1 == dParam.f)),
               angle(Rin(f1 == dParam.f)) / pi * 180);
691         fprintf('Rl   = %+.3f %+.3fj (%.3f /_ %+.2f deg)\n', real(Rl),
               imag(Rl), abs(Rl), angle(Rl) / pi * 180);
692         fprintf('Rout = %+.3f %+.3fj (%.3f /_ %+.2f deg)\n', real(Rout(f1
               == dParam.f)), imag(Rout(f1 == dParam.f)), abs(Rout(f1 ==
               dParam.f)), angle(Rout(f1 == dParam.f)) / pi * 180);
693
694         %fprintf('Rs practical:   %s\n', YesNo{2 - (abs(Rs) < 1 &&
               Sstable(f1 == dParam.f))});
695         %fprintf('Rl practical:   %s\n', YesNo{2 - (abs(Rl) < 1 &&
               Lstable(f1 == dParam.f))});
696         %fprintf('Gain practical: %s\n', YesNo{2 - (MSG(f1 == dParam.f) >
               Gt(f1 == dParam.f))});
697         fprintf('RLi  = %s dB (> 15)\n', num2str(RLidB(f1 == dParam.f),
               '%.2f'));
698         fprintf('RLo  = %s dB (> 15)\n', num2str(RLodB(f1 == dParam.f),
               '%.2f'));
699         fprintf('Ga   = %s dB\n', num2str(GadB(f1 == dParam.f), '%.2f'));
700         fprintf('Gt   = %s dB (> 13)\n', num2str(GtdB(f1 == dParam.f),
               '%.2f'));
701         fprintf('Gp   = %s dB\n', num2str(GpdB(f1 == dParam.f), '%.2f'));
702         fprintf('F    = %s dB (< 0.8)\n', num2str(FdB(f1 == dParam.f),
               '%.3f'));
703         fprintf('\n');
704     end
705
706 %   if (RLidB(f1 == dParam.f) > dParam.RLmin && ...
707 %           RLodB(f1 == dParam.f) > dParam.RLmin && ...
708 %           GtdB(f1 == dParam.f) > dParam.Gmin && ...
709 %           FdB(f1 == dParam.f) < dParam.Fmax)
710 %       fprintf('Specifications met!\n\n');
711 %   else
712 %       fprintf('Specifications NOT met!\n\n');
```

```
%   end
end

% Some test-runs
% Unstable: RL > 15 dB not possible or F < 0.8 -> RL < 1.1 dB

% SerRL = 69.8
% Rs  = -0.208 +0.516j (0.556 /_ 112.00 deg)
% Rl  = 0.356 +0.263j (0.443 /_ 36.40 deg)
% RLi = 15.01 dB (> 15)
% RLo = Inf dB (> 15)
% Ga  = 14.74 dB
% Gt  = 14.74 dB (> 13)
% Gp  = 14.88 dB
% F   = 0.799 dB (< 0.8)

% SerRL = 109
% Rs  = -0.187 +0.443j (0.480 /_ 112.86 deg)
% Rl  = 0.487 +0.152j (0.510 /_ 17.37 deg)
% RLi = 15.01 dB (> 15)
% RLo = Inf dB (> 15)
% Ga  = 13.02 dB
% Gt  = 13.02 dB (> 13)
% Gp  = 13.16 dB
% F   = 0.722 dB (< 0.8)

% SerRL = 93.3
% Rs  = -0.191 +0.469j (0.506 /_ 112.14 deg)
% Rl  = +0.440 +0.185j (0.478 /_ 22.81 deg)
% RLi = 15.00 dB (> 15)
% RLo = Inf dB (> 15)
% Ga  = 13.61 dB
% Gt  = 13.61 dB (> 13)
% Gp  = 13.75 dB
% F   = 0.740 dB (< 0.8)

% Ser-RL = 102.5
% Rs  = -0.191 +0.457j (0.495 /_ 112.72 deg)
% Rl  = +0.468 +0.166j (0.497 /_ 19.52 deg)
% RLi = 15.25 dB (> 15)
% RLo = Inf dB (> 15)
% Ga  = 13.26 dB
% Gt  = 13.26 dB (> 13)
% Gp  = 13.39 dB
% F   = 0.735 dB (< 0.8)

% Ser-RL = 100
% Rs   = -0.191 +0.461j (0.499 /_ 112.57 deg)
% Rin  = -0.307 -0.503j (0.589 /_ -121.45 deg)
% Rl   = +0.461 +0.171j (0.492 /_ 20.36 deg)
% Rout = +0.461 -0.171j (0.492 /_ -20.36 deg)
```

```
% RLi   = 15.22 dB (> 15)
% RLo   = Inf dB (> 15)
% Ga    = 13.35 dB
% Gt    = 13.35 dB (> 13)
% Gp    = 13.49 dB
% F     = 0.737 dB (< 0.8)
```