EXTRACTING BJT SMALL-SIGNAL PARAMETERS
FROM S-PARAMETER MEASUREMENTS

by

Mark Heising

Memorandum No. UCB/ERL M83/20

31 March 1983

(Cover)

EXTRACTING BJT SMALL-SIGNAL PARAMETERS FROM S-PARAMETER MEASUREMENTS

by

Mark Heising

Memorandum No. UCB/ERL M83/20

31 March 1983

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Extracting BJT Small-Signal Parameters from S-Parameter Measurements

Mark Heising

EECS

University of California, Berkeley

ABSTRACT

A method for extracting BJT small-signal parameters from s-parameter measurements is described. Unlike other extraction procedures, this method does require the use of nonlinear numerical methods. Thus the problems of nonconvergence, convergence at local minima, and slow run times, intrinsic to nonlinear optimization programs, are avoided.

The method described here consists of two steps. First, linear least-squares analysis is used to fit the data to network function equations. The resulting fitting coefficients are then solved for the element values of the model.

The method has been implemented for a modified version of the EM3 BJT small-signal model, in a program named EBSS (Extracting Bjt Small-signal parameters from S-parameter measurements). The small-signal parameters derived by this program accurately simulate the device data for frequencies below $f_T/5$.

March 28, 1983

Acknowledgements

The author would like to express his gratitude for the help he received from several people. He would like to thank his research advisor, Professor D. O. Pederson, for his sound advice and firm support. Helpful discussions with Ebrahim Khalily (Hewlett-Packard) are gratefully acknowledged. The assistance of the members of the Computer-Aided-Design Group at the University of California, Berkeley, is also greatly appreciated. The financial support of Tektronix Inc. made possible by Ian Getreu is respectfully acknowledged. Special thanks go to Louis Pengue (Tektronix) for preparing the data which was used in this report.

Extracting BJT Small-Signal Parameters from S-Parameter Measurements

Mark Heising

EECS
University of California, Berkeley

1. Introduction

Computer-aided circuit simulation programs have proven to be an invaluable tool for evaluating circuit designs. Most, if not all, circuit design problems can be diagnosed and corrected before the circuit is actually fabricated. When fabricated circuits don't perform as expected, it is usually because inadequate models or inaccurate model parameters are used in the simulation.

In the case of small-signal analysis, the problem of finding an adequate model and accurate model parameters is equivalent to that of finding a circuit with element values which accurately simulates the n-port characterization of the device. Specifically, this paper addresses itself to the problem of extracting BJT small-signal parameters from 2-port S-parameter measurements in the common-emitter configuration. To minimize the effect of parasitics, the device data was taken on chip.

1.1. Other Methods

One straightforward approach to the problem consists of solving a system of nonlinear equations in terms of the small-signal parameters. This avenue has been independently explored by Ebrahim Khalily [1] and the author. 2-port device measurements taken at a single frequency point provide the known quantities. Equations for these quantities, namely the real and imaginary parts of y_{11}, y_{12}, y_{21} and y_{22} (Khalily's approach), s_{11}, s_{12}, s_{21} and s_{22} (the author's approach), are solved for the small-signal parameters. This necessitates the use of a program which solves nonlinear simultaneous equations.

The problems associated with such programs are well known. The most severe of these is that they do not always converge. There are two reasons this can happen. Programs of this type require the user to make a guess at the solution. If the initial guess is not sufficiently close to a root, the program will diverge, or go to a local minimum. Generally speaking, the more equations there are, the better the initial guess must be. Alternatively, it may be that the system has no solution. As an imperfect model is being used to fit imperfect data, this can be a serious difficulty.

When the program is implemented as described above, it almost always diverges. Khalily found that the incidence of divergence could be substantially reduced by treating re and rc as known quantities. Although this normally allows the program to converge within 10% of the data, the derived parameters typically do not simulate the data very well at frequencies far from the one used in the program. Khalily postulated that part of the problem might lie in the uncertainty incurred in transforming the S-parameters into Y-parameters. This is undoubtedly true. However, the results the author obtained using S-parameters directly yielded no better results.

The crux of the problem is this. The program iterates until a solution is reached that gives less than a 10% error in the Y-parameters. However, this is not equivalent to finding the small-signal parameters to within a 10% error. As the sensitivities of the S-parameters with respect to the various small-signal parameters vary greatly with frequency, the derived model cannot be safely extrapolated to other frequencies.

The customary approach to small-signal parameter extraction is to utilize a nonlinear fitting program. These programs attempt to find the parameters which yield a least-squares fit to the data. This is by definition an optimal set of parameters.

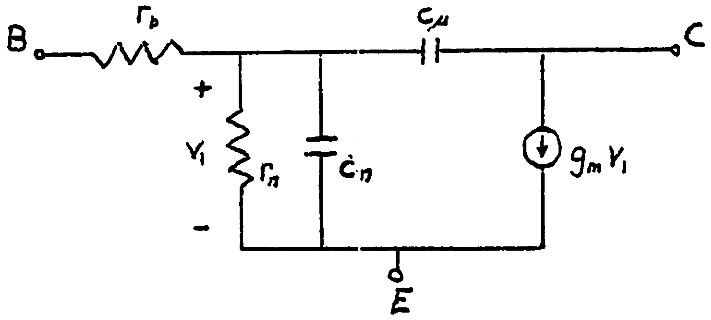
Still, these programs are not without their shortcomings. As with the above method, the user is required to make an initial estimate of the solution. If this estimate is not sufficiently close to the global minimum, the program may diverge or converge at a local minimum. In addition, the equations of interest are quite complex. Run times can therefore be long, even on mainframe computers.

1.2. Overview of the Present Method

The approach taken here can best be summarized by a brief outline.

1) Choose a small signal model.

Example



2) Choose a two-port representation. Solve the admittance matrix of the model for the appropriate network model equations. **Network model equations** are network function equations in terms of the model parameters.

Example

$$y_{21} = \frac{\left[c_u \, r_\pi\right] s - g_m \, r_\pi}{\left[r_b \, r_\pi (c_u + c_\pi)\right] s + r_\pi + r_b}$$

3) Fit network data equations to frequency response measurements of the same two-port. The term **network data equations** is used to to describe the equations which are fitted to the data. These have the same form as network function equations.

Example

$$H(s) = \frac{a_0 + a_1 s}{b_0 + b_1 s} = H_0 \frac{(1 - \frac{s}{z_1})}{(1 - \frac{s}{p_1})}$$

4) The coefficients of the equations from steps 2) and 3) are matched. The resulting system of equations is then solved for the element values.

5) Use SPICE, or some other program, to simulate S-parameter data from the derived model. If the agreement with measurement is not satisfactory, reevaluate the model and/or the method used for fitting the data.

The above method does not offer any advantages over nonlinear optimization in the general case. But for several important models, the above method can yield an extraction program which is much faster than a nonlinear optimization program. The remaining sections cover step by step the procedure outlined above.

To avoid confusing the network model equations with the network data equations, the following conventions are used.

This convention is used to refer to the network model equations.

$$G(s) = \frac{a_0(G) + a_1(G)s + \cdots + a_m(G)s^m}{b_0(G) + b_1(G)s + \cdots + b_n(G)s^n}$$

The fully expanded network model equations can be large and cumbersome. Thus, it is conveniant to have a concise way of referring to these expressions.

The network data equations use a different convention.

$$g(s) = g_o \frac{[1 - \frac{s}{z_1(g)}][1 - \frac{s}{z_2(g)}] \cdots [1 - \frac{s}{z_k(g)}]}{[1 - \frac{s}{p_1(g)}][1 - \frac{s}{p_2(g)}] \cdots [1 - \frac{s}{p_l(g)}]} s^{zp0(g)}$$

where

$zp0(g)$ is an integer which accounts for poles and zeros at $s = 0$

$|z_1| \leq |z_2| \leq |z_3| \cdots$ and

$|p_1| \leq |p_2| \leq |p_3| \cdots$

The above conventions apply only to equations. The term S-parameters (as opposed to s-parameters) will always be used in the text.

2. Small-Signal Model

The choice of which small-signal model to use is crucial to the success of the extraction procedure. This section aims to justify the selection of the modified EM3 small-signal model.

2.1. Criterion and Guidelines for Model Selection

The only firm criterion for selecting a model is that it accurately simulate the S-parameters. Beyond that, there are only guidelines. Still, these guidelines are well worth mentioning. They are as follows:

(1) Use the simplest model that will do the job.

As we go to higher and higher frequencies, the modeling problem becomes increasingly complex. Including high-frequency effects in a relatively low-frequency ac simulation will only increase run time. The situation for transient analysis is not so clear-cut. Ikawa et al.[2] showed that small-signal S-parameters could be used to simulate large-signal switching transients. This was accomplished by integrating the Fourier transforms of the S-parameters over the quasi-static operating points. They found that input rise times of T_r required S-parameter data up to $f_B \approx .35/T_r$. f_B is the system 3-dB frequency. This suggests that, for simulations with 1ns rise times, our model need only be accurate up to 350 MHz.

(2) Each element in the model should have a basis in device physics.

At first this may seem like an unnecessary constraint. Any model, no matter how unorthodox, which accurately simulates the S-parameters is a good model. That is, it meets the criterion for a good model. However, a model with nonphysical elements is probably not going to be useful for more than a handful of devices. Also, the extracted values of nonphysical elements might fluctuate unpredictably over different operating points.

This would render the model useless for transient analysis.

(3) Every element of a model should be extractable.

Extractable means to be determinable by measurement. There are two conditions for extractability. The first requires that a network model equation be sufficiently sensitive to changes in the element value. The second requires that the model parameters not be redundant. These conditions can be restated mathematically.

a) A model element is sufficiently sensitive if for at least one S-parameter in the frequency range of interest,

$$\frac{p_i}{H_{s_j}} \frac{\partial H_{s_j}}{\partial p_i} \gg \varepsilon_{s_j}$$

where,

p_i is a model element.

H_{s_j} is the network model equation of the S-parameter s_j

ε_{s_j} is the relative error associated with the measurement of s_j

If this condition is not met, the determination of the parameter value will be overly sensitive to small changes in the data.

b) A model element is redundant if, for all S-parameters in the frequency range of interest, there exists a vector function $\alpha(\mathbf{p})$ such that,

$$\frac{H_{s_j}(\mathbf{p}) - H'_{s_j}(\alpha(\mathbf{p}))}{H_{s_j}(\mathbf{p})} \ll \varepsilon_{s_j}$$

and

$$rank\,\alpha(\mathbf{p}) < rank\,\mathbf{p}$$

where,

H_{s_j} is the network model equation of the S-parameter s_j

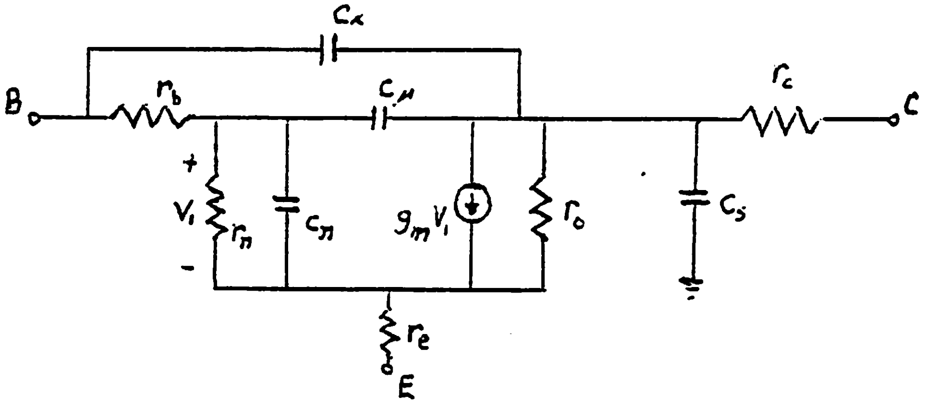
H'_{s_j} is the network model equation of s_j reparameterized.

ε_{s_j} is the relative error associated with the measurement of s_j

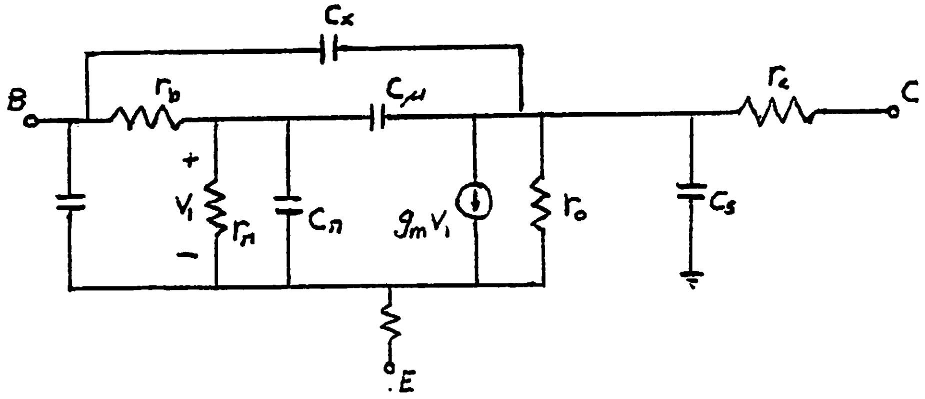
If this condition is not met:an optimal set of parameters can still be found, but the solution will not be unique. That is to say, we can achieve the same modeling accuracy with fewer parameters.

2.2. Modified EM3 model

The original intent of this project was to find a simple method for extracting the small-signal parameters found in the EM3 small-signal model [3].



This model was chosen because it is well-documented and frequently used. It is also fairly simple. However, in the course of this project, it was found that this model could not properly simulate phase even at medium-range frequencies. While introducing a phase delay in g_m could account for the discrepancy in beta, it could not explain the phase problem with input impedance. In addition, at high frequencies, the magnitude of the input impedance was significantly lower than that predicted by the model. All this suggested that a new capacitor, c_x, be introduced to the model.



This is slightly more complicated than the EM3 model. However, because the number of nodes has not been increased, it is expected that the impact on simulation run times will be small.

The physical justification for this capacitance is similar to that used for c_x. The emitter-base junction capacitance c_{je} is distributed across r_b. A first-order approximation to this distributed capacitance is the so-called π network. Moreover, r_b is basically a pinch resistor formed by the emitter diffusion into the base region. But, the largest contribution to c_{je} is from the sidewall region of the base-emitter junction. Since this sidewall capacitance is charged through only a small portion of r_b, much of c_{je} should probably be associated with c_x. It should be noted that the value for this capacitor was found to be an order of magnitude larger than would be expected if it were due only to parasitic bonding capacitances.

All of the model parameters are sufficiently insensitive to small-changes in the data except for r_o. This element is effectively shorted out by r_c in the S-parameter measurement set-up. There are also two sets of redundant parameters. The first set can be reparameterized as,

$$g_m \, r_\pi = \beta_o$$

$$g_m \, r_e = \gamma_o$$

If g_m, r_π and r_e are varied such that β_o and γ_o are constant, very little change will be observed in the S-parameters. For typical values, the sensitivity of the S-parameters to this reparameterization is less than .3% . The other set of redundant parameters can be reparameterized as,

$$c_{te} + (g_m r_c + 1)c_{tc} = c_t$$

where,

c_{te} is the total emitter-base capacitance.

c_{tc} is the total collector-base capacitance.

g_m and c_{tc} are determined elsewhere. That leaves c_{te} and r_c as the redundant parameters. The upshot is that it is difficult to tell the difference between c_{te}, and Miller-effect on c_{tc} through r_c. Actually, when we put typical values into SPICE, we find that s_{22} is able to distinguish c_{te} from r_c at very high frequencies. But at these frequencies, the impedance seen looking into the collector is less than an order of magnitude larger than r_c. It is unrealistic to assume we can measure r_c using s_{22} without taking into account the distribution of c_{tc} and c_s across r_c. This has not been done in the modified EM3 model.

To circumvent the redundancy problem, the program as implemented here requires the user to supply values for r_e and r_c. The choice between r_c and c_{te} is clear, but why not input the collector current and calculate g_m from $g_m = \frac{qI_c}{kT}$? If we account for Gummel-Poon low and high current effects,

$$g_m = \alpha \frac{q\,I_c}{k\,T}$$

where $.5 \le \alpha \le 1$

Unless one is very careful in accounting for temperature and Gummel-Poon effects, the error associated with $\frac{1}{g_m}$ can be as large or larger than r_e.

The program determines the quantity,

$$f00 = \frac{1}{g_m} + r_s$$

So an error in $\frac{1}{g_m}$ which is comparable to r_s will have a catastrophic effect on the determination of r_s. On the other hand, a small error in the value given for r_s has little effect on g_m. If the user decides he would rather specify g_m and obtain r_s, the change can be easily made.

3. Network Model Equations

The reasons for using S-parameters rather than H-parameters are explained in this section. This section also shows how network model equations can be derived from admittance matrices. Finally, the Q-parameters are defined.

3.1. S-parameters vs. H-parameters

Most optimization procedures use H-parameters. These are found from the S-parameters using the following equations.

$$h_{11} = \frac{(1+s_{11})(1+s_{22}) - s_{12}s_{21}}{(1-s_{11})(1+s_{22}) + s_{12}s_{21}} \qquad h_{12} = \frac{2s_{12}}{(1-s_{11})(1+s_{22}) + s_{12}s_{21}}$$

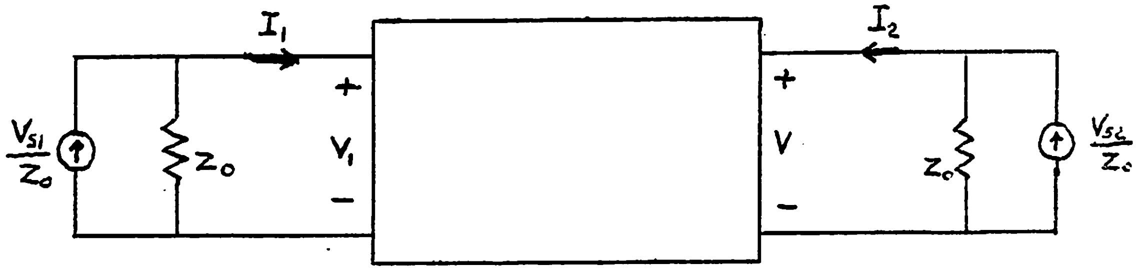
$$h_{22} = \frac{(1-s_{11})(1-s_{22}) - s_{12}s_{21}}{(1-s_{11})(1+s_{22}) + s_{12}s_{21}} \qquad h_{21} = \frac{-2s_{21}}{(1-s_{11})(1+s_{22}) + s_{12}s_{21}}$$

The reason most often given for converting the data is that people have a better intuitive grasp of H-parameters. While in theory one should be able to use any two-port representation he chooses, there are some good reasons for not using H-parameters. First, a computer is extracting the parameters, not a person. A least-squares optimization program can be implemented as easily with S-parameters as with H-parameters. Second, the application of least-squares analysis is valid only if the errors in the data are mutually independent. Yet, the H-parameters all have the same denominator. Thus, their errors are certainly correlated. Third, the relative error in s_{12} is usually larger than that for the other S-parameters. This is because its magnitude is small. Converting to H-parameters causes this larger error to be incorporated in all of the data, instead of just a part of it. Finally, the data should be weighted. The precision of the data points is a function of magnitude, phase, and frequency. A weighting scheme based on the specifications of the network analyzer used to measure the S-parameters is fairly straightforward to implement. It is more difficult to find an appropriate weighting scheme for H-parameters.

For these reasons, the decision was made to work with S-parameters. Regrettably, a scheme using S-parameters directly was not found. There is a two-port representation, however, that makes the problem tractable while avoiding some of the problems associated with H-parameters. These have been called Q-parameters.

3.2. Deriving Network Model Equations from Admittance Matrices

The network model equations for the S-parameters can be defined for the two-port .



$$S_{11} = \left. \frac{V_1 - z_o I_1}{V_1 + z_o I_1} \right|_{v_{s2} = 0} \tag{3.1a}$$

$$S_{21} = \left. \frac{V_2 - z_o I_2}{V_1 + z_o I_1} \right|_{v_{s2} = 0} \tag{3.1b}$$

$$S_{22} = \left. \frac{V_2 - z_o I_2}{V_2 + z_o I_2} \right|_{v_{s1} = 0} \tag{3.1c}$$

$$S_{12} = \frac{V_1 - z_o I_1}{V_2 + z_o I_2}\bigg|_{v_{s1}=0} \qquad (3.1d)$$

Substituting $I_1 = V_{s1}/z_o - V_1/z_o$ and $V_{s1} = V_1 + z_o I_1$ into (3.1a) gives (3.2a). Also, $V_{s2} = 0$ implies $V_2 = -z_o I_2$. Substituting this into (3.1b) gives (3.2b).

$$S_{11} = \left[\frac{2V_1}{V_{s1}} - 1\right]\bigg|_{v_{s2}=0} \qquad (3.2a)$$

$$S_{21} = \frac{2V_2}{V_{s1}}\bigg|_{v_{s2}=0} \qquad (3.2b)$$

Similarly,

$$S_{22} = \left[\frac{2V_2}{V_{s2}} - 1\right]\bigg|_{v_{s1}=0} \qquad (3.2c)$$

$$S_{21} = \frac{2V_1}{V_{s2}}\bigg|_{v_{s1}=0} \qquad (3.2d)$$

Thus, the network model equations for the S-parameters may be derived from an admittance matrix using Cramer's rule. This has been done for the modified EM3 model. The results are in Appendix A.

3.3. Q-parameters

The Q-parameters are defined as follows:

$$Q_{11} = \frac{V_1}{I_1}\bigg|_{v_{s2}=0} \qquad (3.3a)$$

$$Q_{21} = \frac{I_2}{I_1}\bigg|_{v_{s2}=0} \qquad (3.3b)$$

$$Q_{22} = \frac{V_2}{I_2}\bigg|_{u_{s2}=0} \qquad (3.3c)$$

$$Q_{12} = \left. \frac{I_1}{I_2} \right|_{v_{s2}=0} \tag{3.3d}$$

Q_{11} is the impedance seen at port 1 when port 2 is terminated by z_o. Q_{21} is similar to h_{21} with the output terminated in z_o rather than short-circuited. Similar interpretations apply to Q_{12} and Q_{22}.

The Q-parameters can be expressed in terms of the S-parameters.

$$Q_{11}=z_o \frac{1+S_{11}}{1-S_{11}} \tag{3.4a}$$

$$Q_{21}=\frac{S_{21}}{S_{11}-1} \tag{3.4b}$$

$$Q_{22}=z_o \frac{1+S_{22}}{1-S_{22}} \tag{3.4c}$$

$$Q_{12}=\frac{S_{12}}{S_{22}-1} \tag{3.4d}$$

Although most of the arguments made against H-parameters apply to Q-parameters as well, the problems are not as severe with Q-parameters. In particular, the variances of the Q-parameters will be smaller than those for the H-parameters, and the S_{12} problem is avoided. The advantage of Q-parameters over S-parameters is that, in general, the coefficients in the network model equations for Q-parameters depend on fewer device parameters than do those for S-parameters. This leads to a higher degree of independence between the coefficients. This is very useful in solving the system of equations obtained in the fourth step of the method.

3.4. Simplifying the Network Model Equations

A quick look at Appendix A should convince most people that some simplifications are in order. There is one simplification which results in significantly smaller equations with almost no loss in accuracy. It happens that the parameter r_e is important only when it appears multiplied by g_m. By eliminating all terms in which r_e appears without g_m gives us the equations found in Appendix B. All of the above manipulations were carried out on the symbolic algebraic manipulation program MACSYMA [4]. The use of this program is strongly recommended to those whom may wish to explore the subject further.

4. Fitting Polynomials to the Data

Among the easiest equations to fit are polynomials. This section describes how polynomials can be fitted to the data.

4.1. Linear Least-Squares Analysis

There are a number of definitions for best fit. A good general purpose definition says: "Find the values of the constants in the chosen equation that minimize the sum of the squared deviations of the observed values from those predicted by the equation."* One way of doing this is by minimizing the norm of the error function.

$$\left\| f(\mathbf{a}) \right\| = \sum_i \left[\frac{y_i - (a_0 + a_1 x_i + \ldots + a_k x_i^k)}{y_i} \right]^2$$

Solving the system of equations given by

$$\frac{\partial \left\| f(\mathbf{a}) \right\|}{\partial a_i} = 0 \qquad i = 0, 1, 2, \ldots, k$$

for **a**, gives us an optimal set of parameters.

The details can be found in a number of textbooks on statistical methods [5] [6].

*Cuthbert Daniel and Fred S. Wood, *Fitting Equations to Data* ,(New York: Wiley, 1980), p.6.

4.2. Fitting Polynomials to the Data

There are two properties of BJTs which allow us to fit simple polynomials to the data.

1) The zeros of s_{21} and q_{21} are negligible.

   These zeros occur at frequencies beyond f_T. Their location is approximately given by $z_\beta \approx \frac{g_m}{c_\mu}$. Typically, the effect these zeros have on s_{21} and q_{21} *at* $f_T/3$ is about .2% in magnitude and about .3° of phase. (Actually, the program can be made to estimate this zero and then remove its effect from the data. However, this had an almost imperceptible impact on the results.)

2) s_{11}, s_{12} and s_{22} have the same poles as s_{21}. Also, q_{11} has the same poles as q_{21}.

   This is true as long as there are no important parasitic inductances. The simplest way to show this is to note that the network model equations for s_{11}, s_{12}, s_{21} and s_{22} all have the same denominators. As the poles are merely the roots of the denominators, they must have the same poles. The same argument applies to q_{21} and q_{11}.

The first property allows fitting polynomials to $\frac{1}{q_{21}}$ and $\frac{1}{s_{21}}$. The second allows fitting polynomials to $\frac{s_{11}}{s_{21}}$, $\frac{s_{12}}{s_{21}}$, $\frac{s_{22}}{s_{21}}$ and $\frac{q_{11}}{q_{21}}$.

Example

Suppose q_{21} has two significant poles.

Then

$$\frac{1}{q_{21}} = a_0 + a_1 s + a_2 s^2$$

or

$$\left|\frac{1}{q_{21}}\right|^2 = \frac{1}{\beta_o^2}\left(1 + \frac{\omega^2}{p_1^2}\right)\left(1 + \frac{\omega^2}{p_2^2}\right)$$

Suppose q_{11} has one significant zero.

Then

$$\frac{q_{11}}{q_{21}} = a_0 + a_1 s$$

or

$$\left|\frac{q_{11}}{q_{21}}\right| = \frac{Zin_o^2}{\beta_o^2}\left(1 + \frac{\omega^2}{z_1^2}\right)$$

A choice must be made between fitting complex polynomials in terms of s or real polynomials in terms of ω. The program implemented here used primarily the latter. It is an open question whether this is the best choice. The decision to work with magnitudes is based on the fact that they are less sensitive to parasitics and higher-order poles and zeros. If the major poles and zeros are properly located using magnitude, the phase must fall into place.

4.3. Network Functions Selected for Fitting

The following functions were chosen because of their sensitivity to model parameters and their relative insensitivity to perturbations in the data. The quantities on the left-hand-side represent data. The quantities on the right-hand-side are polynomial functions of ω. The polynomials are factored to emphasize the coefficients used in the next section. Once a coefficient is determined it may appear on the left-hand-side in subsequent fitting equations.

$$\left|\frac{1}{q_{21}}\right|^2 = \frac{1}{(q_{21_0})^2}\left[1+\left(\frac{\omega}{p_1(q_{21})}\right)^2\right]\left[1+\left(\frac{\omega}{p_{eff}}\right)^2\right] \qquad (4.1)$$

q_{21_0} is beta at low-frequencies and p_{eff} is an effective pole which allows for the effect of higher order poles on the magnitude of q_{21}. It is not used anywhere else.

$$\left|\frac{q_{21_0}\, q_{11}}{q_{21}}\right|^2 = (q_{11_0})^2\left[1+\left(\frac{\omega}{z_1(q_{11})}\right)^2\right] \qquad (4.2)$$

q_{11_0} is the input impedance at low-frequencies.

$$\left|\frac{s_{21_0}\, s_{12}}{\omega\, s_{21}}\right|^2 = (s_{12_0})^2\left[1+\left(\frac{\omega}{z_1(s_{12})}\right)^2\right] \qquad (4.3)$$

s_{21_0} is already known at this point from the relation $s_{21_0} = -2 z_0\, q_{21_0}$.

$$\mathrm{Im}\left[\frac{q_{21_0}}{\left(1-\dfrac{j\,\omega}{p_1(q_{21})}\right) q_{21}}\right] = -\left[\frac{1}{p_2(q_{21})}+\frac{1}{p_3(q_{21})}\right]\omega \qquad (4.4)$$

This complex quantity is used to find c_z. There are two reasons for using a complex quantity here. First, p_2 and p_3 , usually occur at frequencies too high to be determined by magnitude. Second, the primary effect of c_z is on phase.

Because the data has not been weighted explicitly, it must be weighted implicitly by selecting the the frequency range over which the above equations are to be fitted. The following table gives the suggested frequency ranges in terms of the lowest available frequency, LF, and the unity gain frequency, f_T. The lowest available frequency is a function of the network analyzer used to measure the S-parameters. These frequency ranges are determined empirically and should be used only as a guideline.

Eqn.	Frequency Range
4.1	$LF \quad f_T$
4.2	$LF \quad f_T/3$
4.3	$LF \quad f_T/3$
4.4	$f_T/10 \quad f_T/3$

5. Matching and Solving the Coefficients

In this section, the correspondence between the network model equations and the network data equations is specified. An efficient method for solving the resulting system of equations is subsequently delineated.

5.1. Matching the Coefficients

There are two sets of coefficients. One set consists of the coefficients in the network model equations derived in Section 3 and found in Appendix B. The other set consists of the coefficients found by fitting the data to polynomials. The task of matching the coefficients found in the previous two sections would be trivial, but for one thing. The order of the polynomials in the network model equations is sometimes larger than that of the corresponding network data equation. That is because some of the poles and zeros predicted by the network model equations occur at frequencies at which the model breaks down. Thus, they cannot be found by fitting the data. We could find the roots of the network model equations and match them with the appropriate poles or zeros, but this results in nonlinear equations which must in turn be solved for the small-signal parameters. Rather than resorting to nonlinear numerical methods, certain approximations are made that result in linear equations which are easily inverted.

The following network model coefficients can be identified with the appropriate poles, zeros or scale factors by inspection. The symbol '::' should be read as "corresponds to".

$$q_{11_0} :: a_0(Q_{11}) = r_b + (g_m r_e + 1) r_\pi \tag{5.1}$$

$$q_{21_0} :: a_0(Q_{21}) = g_m r_\pi \tag{5.2}$$

$$s_{12_o} :: a_1(S_{12}) = 2z_o\,(c_{tc}\,(g_m r_e + 1) r_\pi + c_x r_b) \tag{5.3}$$

$$z_1(s_{12}) :: -\frac{a_1(S_{12})}{a_2(S_{12})} = -\frac{r_b + (g_m r_e + 1) r_\pi}{[(c_z + c_x) c_\mu g_m r_e + (c_\mu + c_\pi) c_x] r_b r_\pi} \tag{5.4}$$

Approximations are made for the following matches.

$$p_1(q_{21}) :: root_1[\,den\,(Q_{21})\,] \tag{5.5a}$$

where,

$den\,(Q_{21})$ is the denominator of Q_{21}, (see Appendix B)

$root_1$ is the root with the smallest magnitude

The expanded expression for $root_1(den\,(Q_{21})$ is too complex to be useful. To a good approximation, the effect of c_s, r_b and r_e on this root is negligible. (Typically less than .1% .)

Thus,

$$root_1[\,den\,(Q_{21})\,] \approx root_1[\,den\,(Q_{21})|_{c_s=0, r_b=0, r_e=0}\,] \tag{5.5b}$$

$$= [\,c_{tc}\,c_{te}\,(r_c + z_o) r_\pi\,] s^2$$

$$+ [\,(\,c_{te}\,(g_m\,(r_c + z_o) + 1) + c_{te}\,) r_\pi + c_{tc}\,(r_c + z_o)\,] s + 1$$

We could solve this directly using the quadratic formula, but further approximations must be made which result in a linear expression.

We have an equation in the form of,

$$a x^2 + b x + 1 = \left(1 - \frac{x}{x_1}\right)\left(1 - \frac{x}{x_2}\right) = 0$$

$$x = \frac{-b \pm b\sqrt{1 - \frac{4a}{b^2}}}{2a}$$

If $b^2 \gg 4a$,

$$x \approx \frac{-b \pm b\,(1 - \frac{2a}{b^2})}{2a}$$

then,

$$x_1 \approx -\frac{1}{b} \qquad x_2 \approx -\frac{b}{a}$$

But,

$$-\left[\frac{1}{x_1} + \frac{1}{x_2}\right] = b$$

so a better approximation to x_1 is,

$$x_1 \approx \frac{1}{-b + \frac{a}{b}}$$

We can rewrite this as,

$$\frac{1}{x_1} + \kappa \approx b$$

where $\kappa = \frac{a}{b}$, is a small correction factor

From Equation (5.5b),

$$[\,(\,c_{tc}\,(g_m\,(r_c\, + z_o)\, + 1) + c_{te}\,)r_\pi + c_{tc}\,(r_c + z_o)\,]^2 \gg 4[\,c_{tc}\,c_{te}\,(r_c + z_o)r_\pi\,]$$

This condition is satisfied as long as the dominant pole of q_{21} is far from the non-dominant poles.

So,

$$\frac{1}{p_1(q_{21})} + \kappa_p :: -\left(c_{tc}\left(g_m\left(r_c + z_o\right) + 1\right) + c_{te} \right) r_\pi + c_{tc}\left(r_c + z_o\right) \qquad (5.5c)$$

where,

$$\kappa_p = \frac{\left(c_{tc}\left(g_m\left(r_c + z_o\right) + 1\right) + c_{te}\right) r_\pi + c_{tc}\left(r_c + z_o\right)}{c_{tc}\, c_{te}\left(r_c + z_o\right) r_\pi}$$

Similar reasoning is used to match $z_1(q_{11})$.

$$z_1(q_{11}) :: root_1[num\,(Q_{11})] \qquad (5.6a)$$

where,

$num\,(Q_{11})$ is the numerator of Q_{11}, (see Appendix B)

To a good approximation, the effect of c_z on this root is negligible.

Thus,

$$root_1[num\,(Q_{11})] \approx root_1[num\,(Q_{11})|_{c_z=0}] \qquad (5.6b)$$

Then,

$$\frac{1}{z_1(q_{11})} + \kappa_z = \qquad (5.6c)$$

$$-\frac{r_b + (g_m r_e + 1) r_\pi}{r_\pi (r_c + z_o)[c_{tc}(g_m r_e + 1) + c_\mu g_m r_b] + r_b (r_c + z_o) c_{tc} + r_b r_\pi [c_\mu (g_m r_e + 1) + c_\pi]}$$

where,

$$\kappa_z = \frac{r_\pi (r_c + z_o)[c_{tc}(g_m r_e + 1) + c_\mu g_m r_b] + r_b (r_c + z_o) c_{tc} + r_b r_\pi [c_\mu (g_m r_e + 1) + c_\pi]}{r_b (r_c + z_o) r_\pi [(c_x + c_z) c_\mu g_m r_e + c_{tc} c_\pi + c_\mu c_x]}$$

Finally, the quantity from Equation (4.4) is matched.

$$\frac{1}{p_1(q_{21})}\left[\frac{1}{p_2(q_{21})}+\frac{1}{p_3(q_{21})}\right] :: b_2(Q_{21}) - \frac{1}{root_2(Q_{21})\, root_3(Q_{21})} \qquad (5.7a)$$

Now,

$$b_2(Q_{21}) = \frac{1}{root_1(Q_{21})\, root_2(Q_{21})} + \frac{1}{root_1(Q_{21})\, root_3(Q_{21})} + \frac{1}{root_2(Q_{21})\, root_3(Q_{21})}$$

but,

$$\frac{1}{root_1(Q_{21})} \gg \frac{1}{root_2(Q_{21})},\ \frac{1}{root_3(Q_{21})}$$

so,

$$\frac{1}{p_1(q_{21})}\left[\frac{1}{p_2(q_{21})}+\frac{1}{p_3(q_{21})}\right] :: b_2(Q_{21}) \qquad (5.7b)$$

See Appendix B for the expanded expression of $b_2(Q_{21})$.

There are now seven equations for eight unknowns. These can be solved for all the small-signal parameters except for c_s. Network model coefficients were found which were sensitive to c_s but they were also sensitive to r_c. Since the value supplied by the user for r_c might not be very accurate, a somewhat different scheme was used to find c_s. This is explained in the next subsection.

5.2. Solving for the Element Values

The equations in Subsection 5.1. have a useful characteristic. Though the network model coefficients may be functions of several small-signal parameters, they are generally strong functions of only a few parameters. This property allows us to set up the equations for the small-signal parameters in a selfconverging loop. No initial estimates of the solution need be supplied by the user. To get started, the program supplies its own guess.

Consider,

$$Zin_o = r_b + \beta_o \left(r_e + \frac{1}{g_m}\right) + r_e \tag{5.9a}$$

This suggests,

$$Zin(j\omega) \approx r_b + \beta(j\omega)\left(r_e + \frac{1}{g_m}\right) + r_e \tag{5.9b}$$

If we approximate $Zin(j\omega)$ with a 1 pole, 1 zero f.inction, and $\beta(j\omega)$ with a 1 pole function, we can solve (5.9a) and (5.9b) for $\frac{1}{g_m} + r_e$.

$$foo = \frac{1}{g_m} + r_e = \frac{Zin_o}{\beta_o}\left[1 - \frac{zero}{pole}\right] + \kappa_f \tag{5.9c}$$

where κ_f is an error term.

Substitute the following expressions:

q_{11_o} for Zin_o

q_{21_o} for β_o

$p_1(q_{21})$ for $pole$

$z_1(s_{12})$ for $zero$

Using the corresponding network model coefficients found in equations (5.1),(5.2),(5.6c) and (5.5c), we can find an expression for κ_f in terms of the small-signal parameters. This rather involved expression can be found in the program listing under Subroutine LOOP.

We now have expressions for g_m and r_b.

$$g_m = \frac{1}{foo \ - r_e} \qquad (5.10a)$$

$$r_b = q_{11_o} - q_{21_o} foo \ - r_e \qquad (5.10b)$$

From (5.1) and (5.10a),

$$r_\pi = \frac{q_{21_o}}{g_m} \qquad (5.10c)$$

From (5.3),

$$c_{tc} = \frac{s_{12_o} + c_\mu r_b}{q_{11_o}} \qquad (5.10d)$$

Since,

$$c_{tc} \, q_{11_o} \gg c_\mu r_b$$

the value for c_μ is not important. It is essentially another correction factor which is found by iteration.

From (5.5c),

$$c_{te} = -\frac{p_1(q_{21})[\, c_{tc} \, (1 + g_m \, (r_c \, + z_o))\, r_\pi + c_{tc} \, (r_c \, + z_o)\,] + 1}{p_1(q_{21})\, r_\pi} \qquad (5.10e)$$

The remaining expressions are long but straightforward to derive.

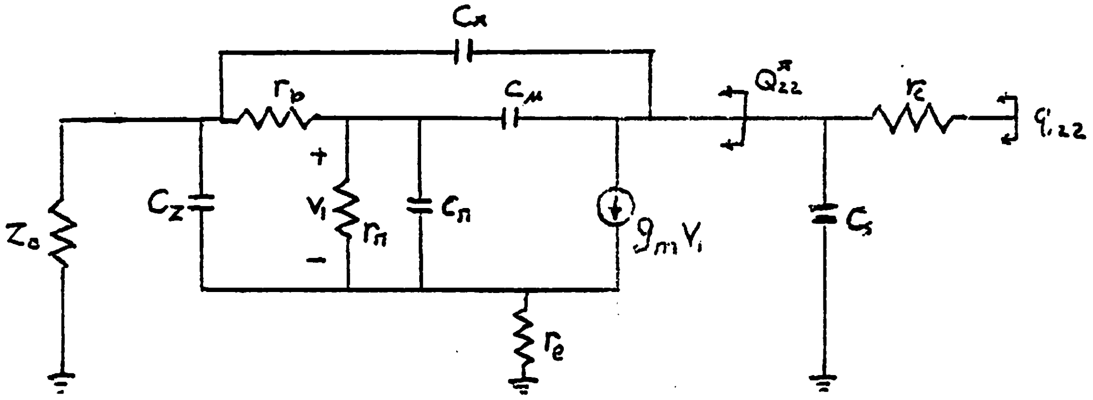
Equation (5.4) is solved for c_x giving (5.10f)

Equation (5.7b) is solved for c_z giving (5.10g)

The expanded expressions for (5.10f) and (5.10g) can be found in Appendix E in the program listing for Subroutine LOOP.

The equations are solved with the correction terms κ_p, κ_z, and κ_f set to zero. These corrections are calculated and applied to $p_1(q_{21})$, $z_1(q_{11})$ and foo, respectively. By iterating through the equations, the program finds successively better estimates for the element values, which converge on their final values asymptotically. In practice, 3 to 5 iterations are all that is required.

The following diagram is used to find c_s.



$$\frac{1}{q_{22} - r_c} = \frac{1}{Q^*_{22}} + s\, c_s \qquad (5.10h)$$

The left-hand-side is determined by measurement. The quantity, Q^*_{22} is a complicated expression in terms of the other small-signal parameters. The values for these parameters are supplied by Subroutine LOOP. The expanded expression for Q^*_{22} can be found in Appendix E in the program listing for Subroutine CCS. Equation (5.8) is solved for c_s at several frequency points. The average value is taken and returned to Subroutine LOOP. If this is done only for frequencies such that $|\, Q_{22} \,| \gg r_c$, the result will not be sensitive to the value given for r_c. Frequencies at which $s_{22} \approx 1$ should also not be used, as they will lead to inaccurate values for q_{22}.

6. Results

Simulated and measured S-parameters are compared in the following plots. The S-parameter measurements were taken with TECAP on a 600MHz device. SPICE was used to simulate the S-parameters. Discrepancies between measurement and simulation at low-frequencies are due primarily to measurement error. The phase of s_{12} is especially inaccurate. Discrepancies at high-frequencies are mainly due to the effect of the substrate resistance. The SPICE files used in the simulation can be found in Appendix F.

The data is summarized by taking the average of the absolute values of the errors.

$$aae\,(magnitude) = \frac{1}{n} \cdot \sum_{i=1}^{n} \left| \frac{qm_i - qs_i}{qm_i} \right|$$

$$aae\,(phase) = \frac{1}{n} \sum_{i=1}^{n} \left| qm_i - qs_i \right|$$

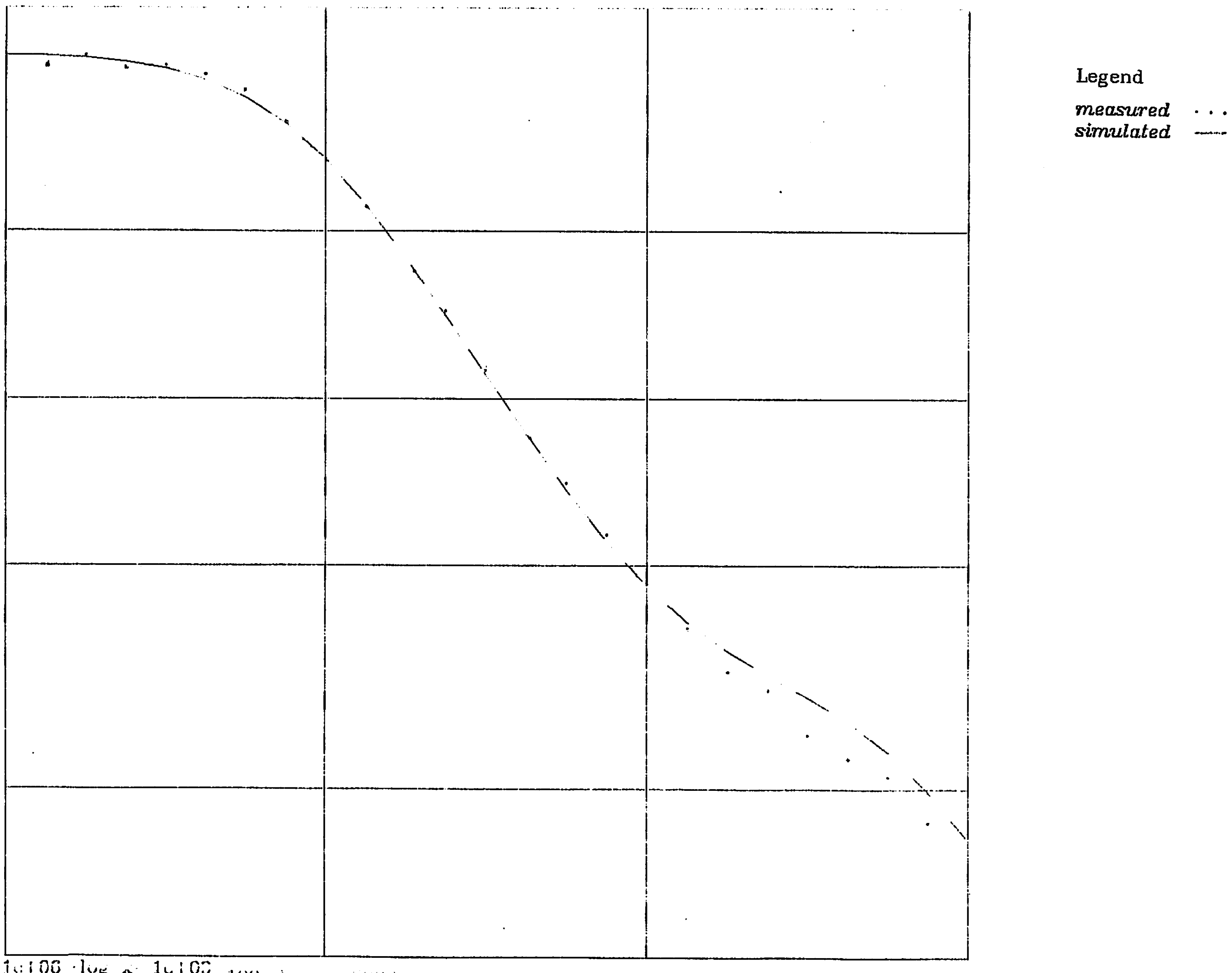
qm_i is the measured quantity at the ith frequency point. qs_i is the simulated quantity at the ith frequency point.

The following table gives the aae for frequencies from 5Mhz to f_T and from 5MHz to $f_T/4$.

Average of the Absolute Errors		
Quantity	f_T	$f_T/4$
mag. q_{11}	4.7%	1.9%
mag. q_{21}	1.8%	1.3%
mag. s_{12}	6.5%	5.6%
mag. q_{22}	7.6%	3.2%
phase q_{11}	4.7°	1.9°
phase q_{21}	3.1°	.9°
phase s_{12}	10.3°	7.0°
phase q_{22}	8.7°	4.5°

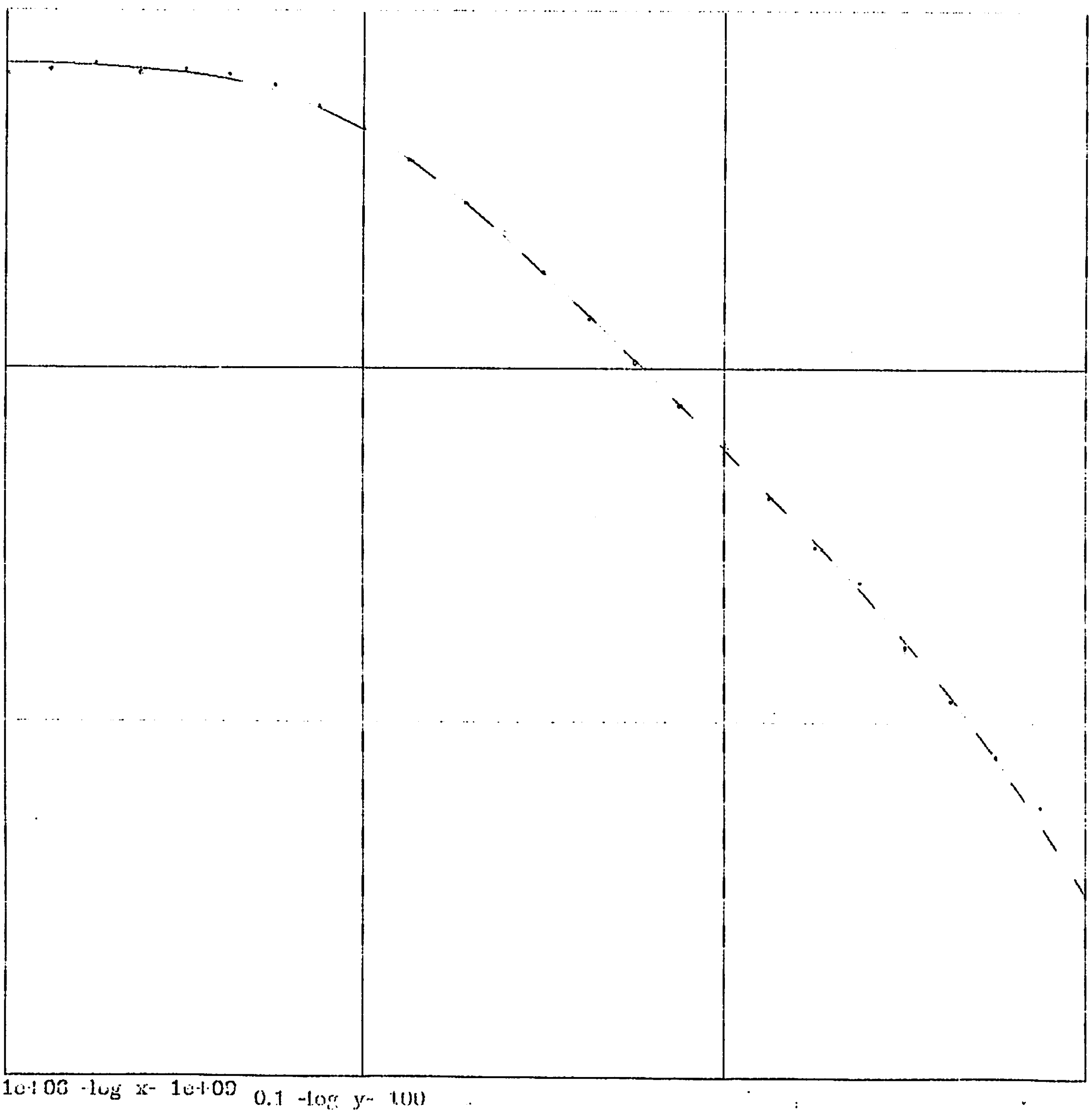
marcos:Tue Mar 22 17:23:02 1983
vplot

magnitude of q11

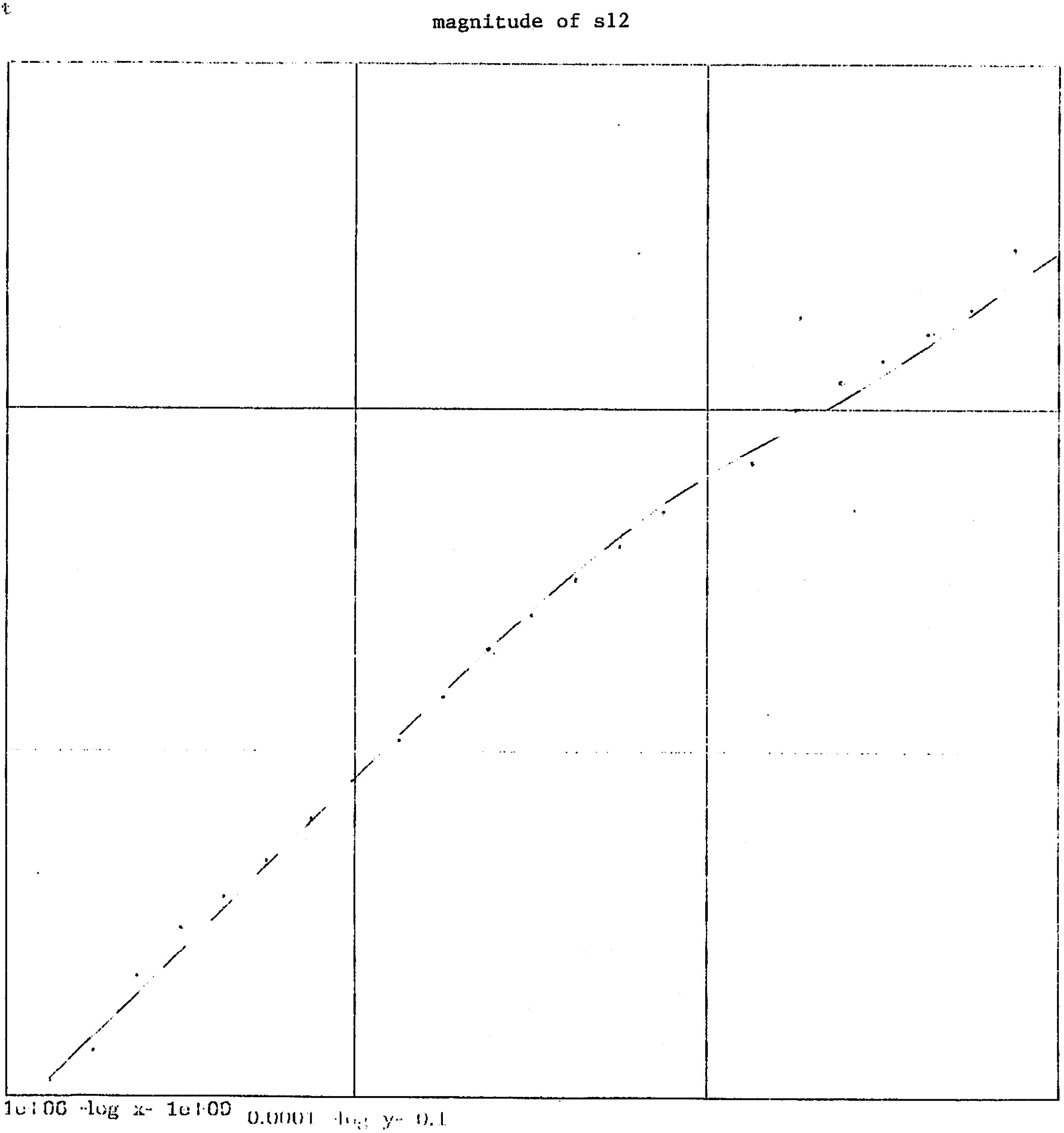


marcos:Tue Mar 22 17:23:49 1983

magnitude of q21

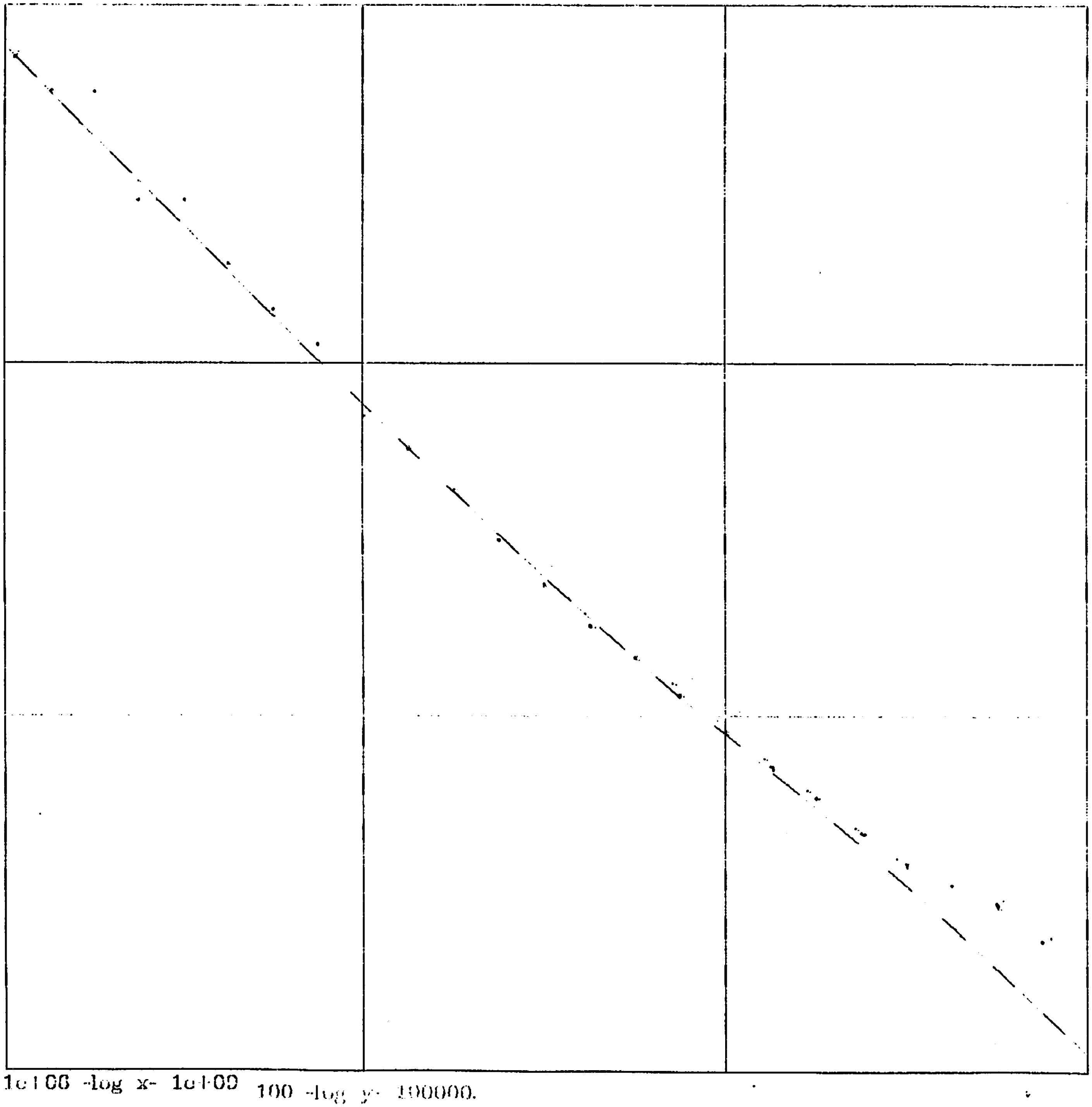


marcos:Tue Mar 22 17:24:37 1983
vplot



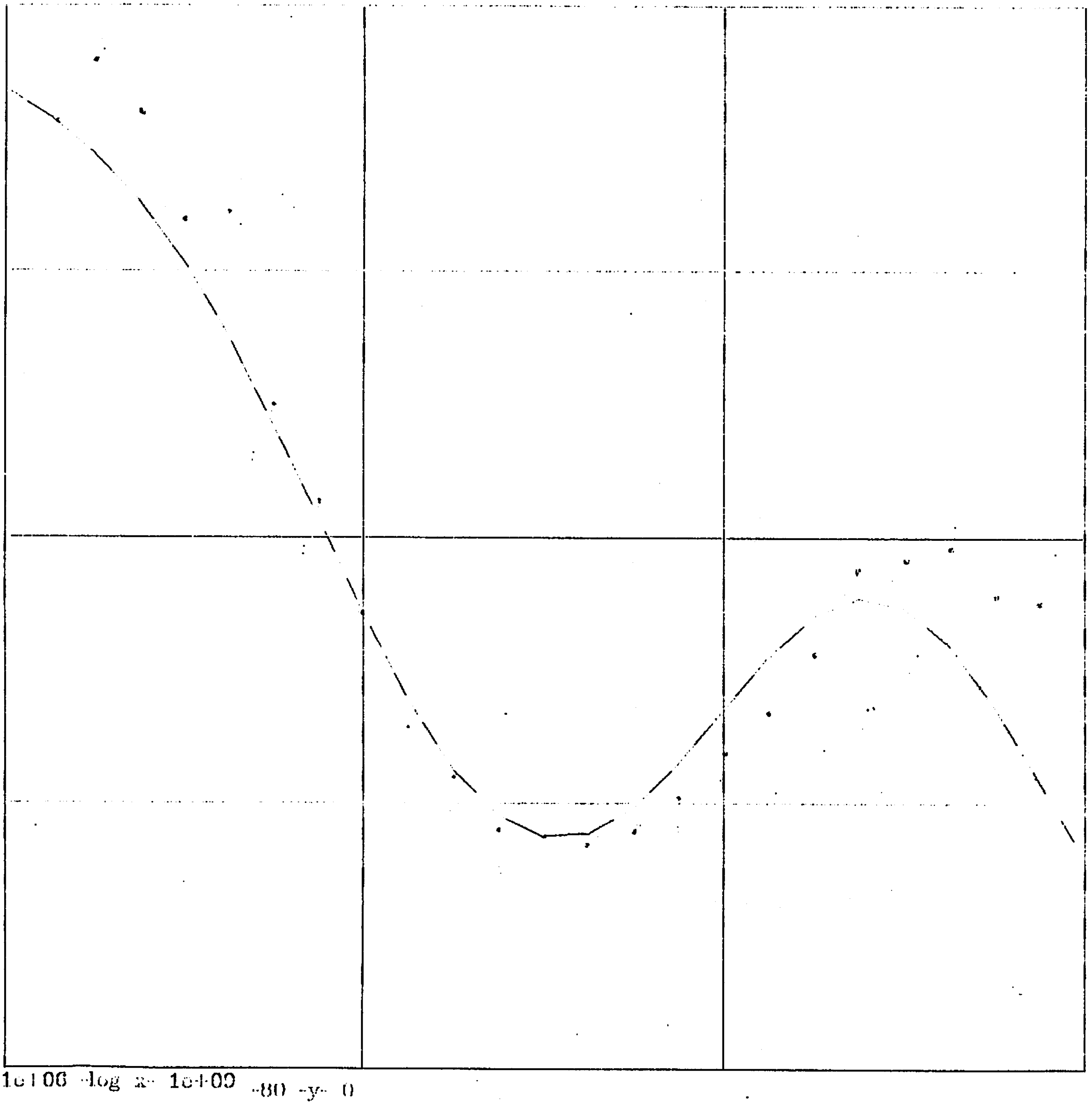
marcos:Tue Mar 22 17:25:26 1983
qplot

magnitude of q22



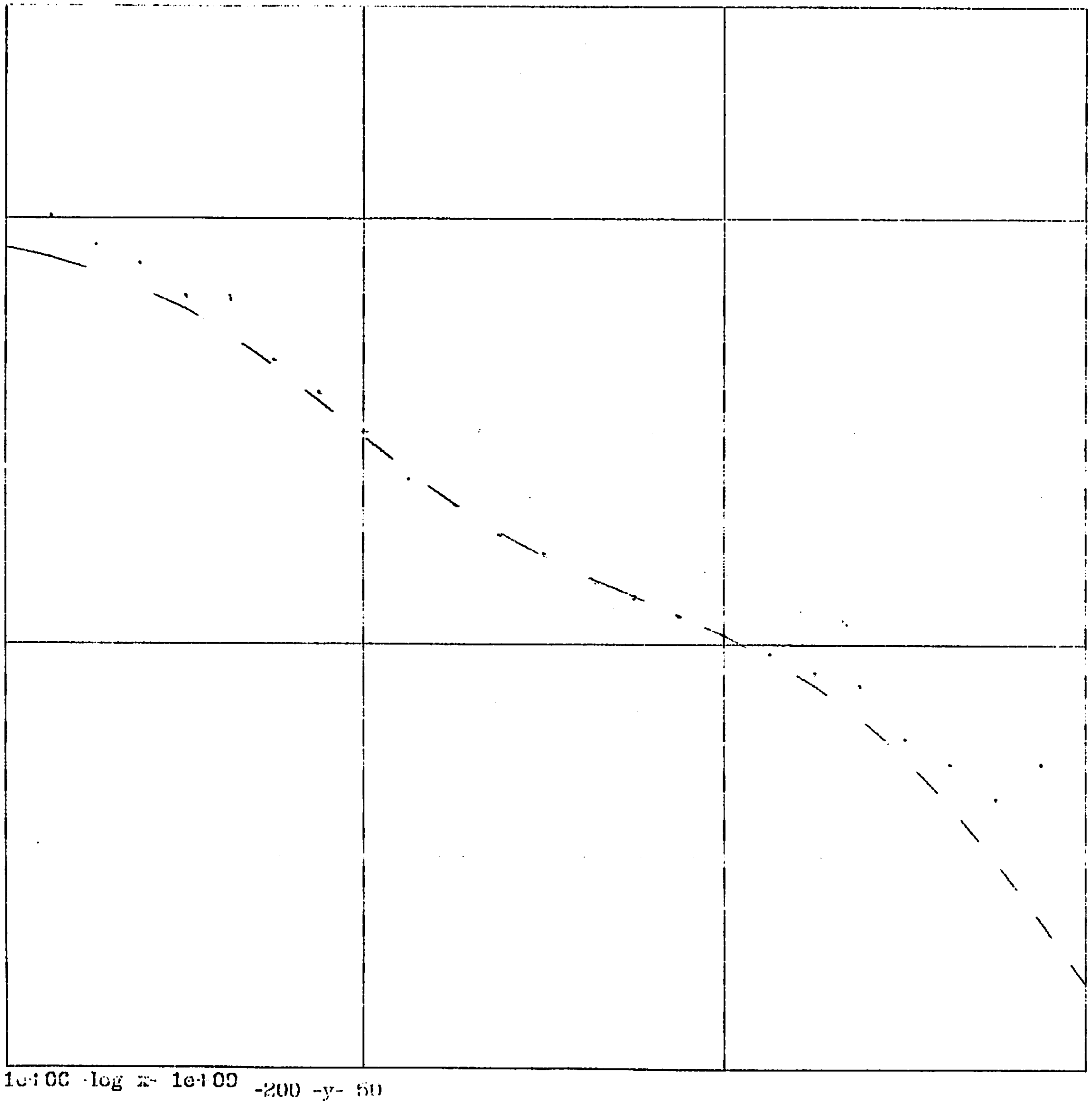
marcos:Tue Mar 22 17:35:02 1983
vplot

phase of q11



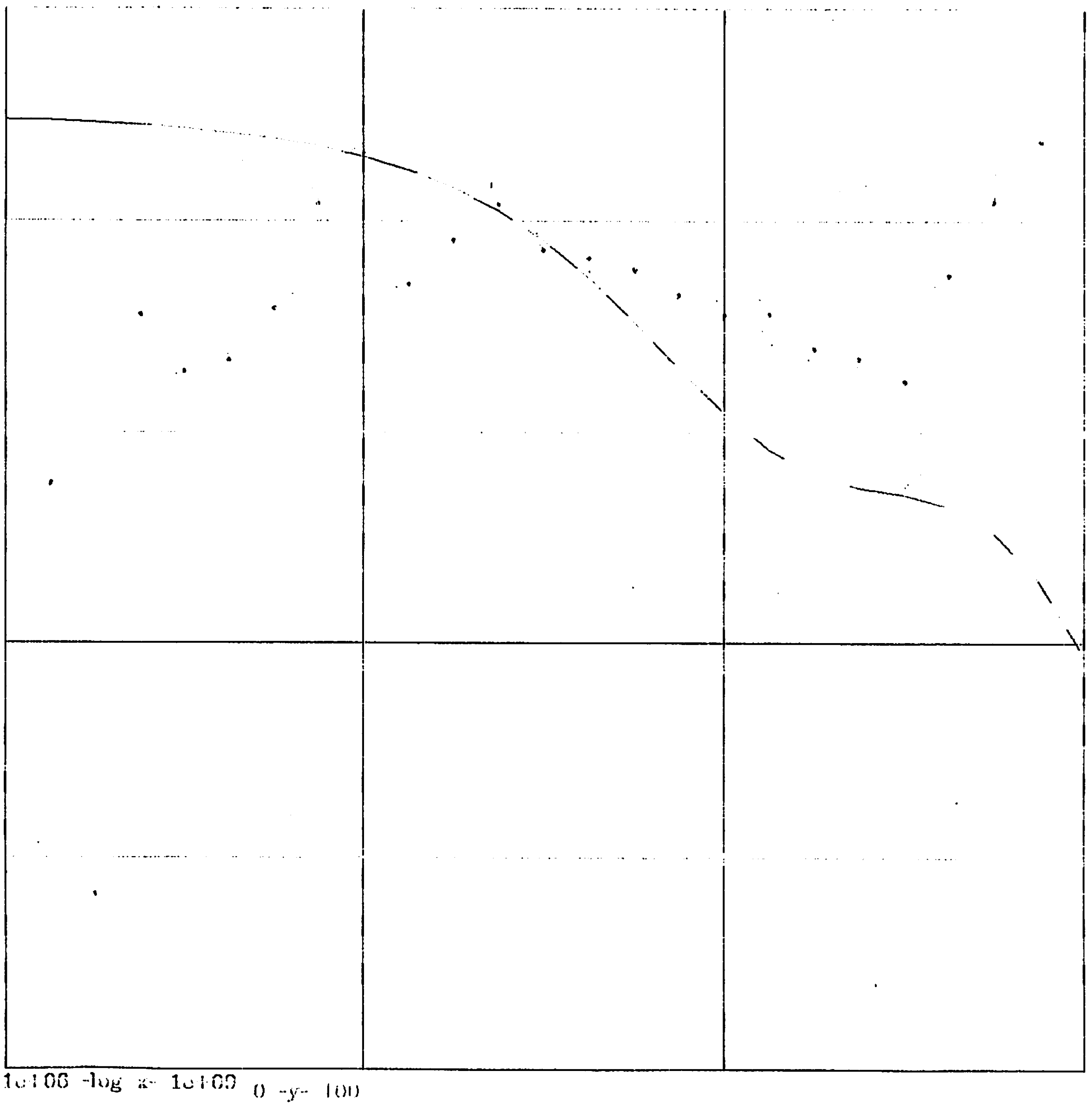
marcos:Tue Mar 23 18:00:48 1993
vplot

phase of q21



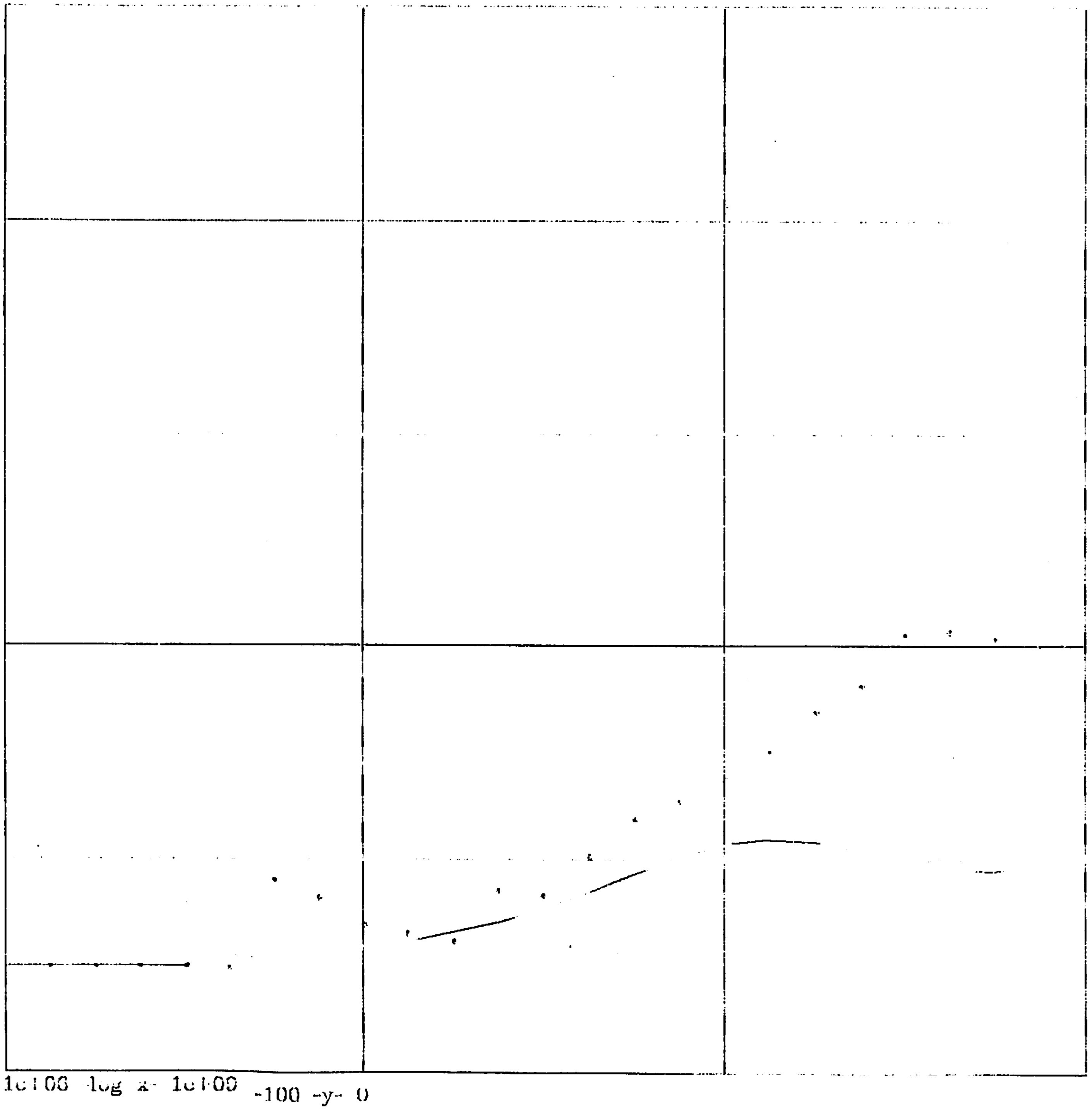
marcos:Tue Mar 22 17:59:01 1983
vplot

phase of s12



marcos:Tue Mar 22 18:02:22 1983
vplot

phase of q22



Simulated and measured S-parameters are compared in tables on the following pages. The S-parameter measurements were taken by Louis Pengue of Tektronix on a 7GHz device. The tables also compare simulations from device parameters produced by SCALE to measurements. SCALE is a program developed by Tektronix Inc. which produces SPICE parameters from processing parameters and device geometry.

This table gives the aae for SCALE and EBSS at two operating points.

Average of the Absolute Errors				
Quantity	$I_c = 2ma.$		$I_c = 5ma.$	
	SCALE	EBSS	SCALE	EBSS
mag. q_{11}	14.5%	4.8%	37.8%	4.5%
mag. q_{21}	6.1%	2.8%	4.2%	1.9%
mag. s_{12}	4.3%	4.7%	9.7%	4.7%
mag. q_{22}	15.9%	3.7%	14.5%	3.9%
phase q_{11}	15.9°	3.7°	16.4°	1.5°
phase q_{21}	4.2°	2.9°	4.8°	2.9°
phase s_{12}	4.7°	6.6°	6.8°	5.6°
phase q_{22}	7.5°	2.7°	7.4°	3.0°

Magnitude of q_{11} , $I_c = 2ma$, $V_{ce} = 3V$					
Freq.(MHz)	Measured	SCALE	Δ %	EBBS	Δ %
100	517.8	488.2	-5.7	553.4	6.9
200	310.2	270.3	-12.8	301.5	-2.8
300	206.7	188.1	-9.0	206.4	-0.1
400	147.7	146.7	-0.6	157.9	6.9
500	125.1	122.5	-2.0	128.8	3.0
600	114.5	106.9	-6.6	109.7	-4.1
700	101.4	96.3	-5.1	96.3	-5.0
800	88.6	88.6	0.0	86.5	-2.4
900	81.3	83.0	2.1	79.0	-2.8
1000	76.3	78.7	3.2	73.2	-4.1
1100	72.9	75.4	3.4	68.6	-5.9
1200	65.1	72.8	11.8	64.9	-0.4
1300	57.7	70.7	22.4	61.8	7.1
1400	53.7	68.9	28.4	59.3	10.4
1500	52.9	67.5	27.8	57.1	8.1
1600	52.8	66.4	25.7	55.3	4.8
1700	52.0	65.4	25.7	53.8	3.4
1800	50.0	64.6	29.2	52.5	5.0
1900	47.5	63.9	34.5	51.3	8.1
2000	47.5	63.3	33.1	50.3	5.9

Phase of q_{11} , $I_c = 2ma$, $V_{ce} = 3V$					
Freq.(MHz)	Measured	SCALE	Δ deg	EBBS	Δ deg
100	-54.3	-56.8	-2.5	-60.4	-6.1
200	-67.5	-63.9	3.6	-69.6	-2.2
300	-71.6	-63.1	8.6	-71.0	0.6
400	-71.8	-60.3	11.5	-70.3	1.5
500	-71.1	-56.9	14.2	-68.8	2.4
600	-67.9	-53.4	14.4	-66.9	1.0
700	-66.2	-50.1	16.1	-64.8	1.4
800	-63.6	-47.0	16.7	-62.8	0.8
900	-61.1	-44.1	17.0	-60.8	0.3
1000	-58.7	-41.4	17.3	-58.9	-0.2
1100	-55.9	-39.0	16.9	-57.1	-1.1
1200	-55.3	-36.8	18.5	-55.4	-0.0
1300	-54.8	-34.8	20.0	-53.8	1.0
1400	-54.9	-33.0	21.9	-52.3	2.7
1500	-52.8	-31.4	21.4	-50.9	1.9
1600	-50.8	-29.9	21.0	-49.6	1.2
1700	-49.9	-28.5	21.4	-48.5	1.4
1800	-49.0	-27.3	21.7	-47.4	1.6
1900	-47.5	-26.2	21.4	-46.5	1.1
2000	-46.3	-25.1	21.2	-45.6	0.7

Magnitude of s_{12} , I_c = $2ma$, V_{ce} = $3V$					
Freq.(MHz)	Measured	SCALE	Δ %	EBBS	Δ %
100	0.0200	0.0153	-23.4	0.0138	-30.9
200	0.0270	0.0294	8.9	0.0270	-0.0
300	0.0370	0.0415	12.2	0.0390	5.3
400	0.0510	0.0516	1.2	0.0495	-2.9
500	0.0580	0.0600	3.4	0.0586	1.0
600	0.0640	0.0670	4.6	0.0663	3.6
700	0.0720	0.0730	1.4	0.0729	1.2
800	0.0800	0.0784	-2.0	0.0785	-1.9
900	0.0860	0.0834	-3.1	0.0833	-3.1
1000	0.0900	0.0880	-2.2	0.0875	-2.7
1100	0.0920	0.0925	0.6	0.0913	-0.7
1200	0.0950	0.0969	2.0	0.0947	-0.3
1300	0.0990	0.1012	2.2	0.0979	-1.2
1400	0.1050	0.1055	0.5	0.1008	-4.0
1500	0.1050	0.1097	4.5	0.1035	-1.4
1600	0.1080	0.1139	5.5	0.1062	-1.7
1700	0.1150	0.1182	2.8	0.1087	-5.5
1800	0.1220	0.1224	0.3	0.1111	-8.9
1900	0.1260	0.1266	0.5	0.1135	-9.9
2000	0.1260	0.1308	3.8	0.1158	-8.1

Phase of s_{12} , I_c = $2ma$, V_{ce} = $3V$					
Freq.(MHz)	Measured	SCALE	Δ deg	EBBS	Δ deg
100	84.0	82.4	-1.6	83.3	-0.8
200	82.0	75.5	-6.5	76.8	-5.2
300	69.0	69.7	0.7	70.8	1.7
400	68.0	65.0	-3.0	65.3	-2.7
500	69.0	61.4	-7.6	60.6	-8.4
600	66.0	58.7	-7.3	56.5	-9.5
700	59.0	56.7	-2.3	53.0	-6.0
800	53.0	55.3	2.3	50.0	-3.0
900	52.0	54.3	2.3	47.4	-4.6
1000	51.0	53.6	2.6	45.2	-5.8
1100	50.0	53.1	3.1	43.3	-6.7
1200	47.0	52.8	5.8	41.7	-5.3
1300	46.0	52.5	6.5	40.3	-5.7
1400	47.0	52.3	5.3	39.1	-7.9
1500	50.0	52.1	2.1	38.0	-12.0
1600	50.0	52.0	2.0	37.0	-13.0
1700	46.0	51.9	5.9	36.1	-9.9
1800	43.0	51.8	8.8	35.3	-7.7
1900	42.0	51.6	9.6	34.5	-7.5
2000	42.0	51.5	9.5	33.8	-8.2

Magnitude of q_{21}, $I_c = 2ma$, $V_{ce} = 3V$					
Freq.(MHz)	Measured	SCALE	Δ %	EBBS	Δ %
100	33.42	31.39	-6.1	33.95	1.6
200	19.98	17.11	-14.3	18.38	-8.0
300	12.58	11.61	-7.7	12.46	-1.0
400	8.80	8.77	-0.3	9.40	6.8
500	7.31	7.05	-3.6	7.53	3.1
600	6.43	5.89	-8.4	6.29	-2.2
700	5.56	5.06	-9.0	5.39	-3.0
800	4.74	4.43	-6.5	4.72	-0.5
900	4.24	3.95	-6.8	4.19	-1.0
1000	3.87	3.56	-8.0	3.77	-2.5
1100	3.58	3.24	-9.4	3.43	-4.2
1200	3.15	2.98	-5.5	3.14	-0.4
1300	2.82	2.76	-2.2	2.90	2.8
1400	2.60	2.57	-1.2	2.69	3.5
1500	2.50	2.40	-3.9	2.51	0.3
1600	2.43	2.26	-7.1	2.35	-3.4
1700	2.32	2.13	-8.1	2.21	-4.8
1800	2.16	2.02	-6.4	2.08	-3.4
1900	1.99	1.92	-3.5	1.97	-0.8
2000	1.91	1.83	-4.1	1.87	-1.9

Phase of q_{21}, $I_c = 2ma$, $V_{ce} = 3V$					
Freq.(MHz)	Measured	SCALE	Δ deg	EBBS	Δ deg
100	-60.1	-63.8	-3.7	-65.2	-5.1
200	-77.5	-77.7	-0.2	-79.2	-1.7
300	-87.6	-83.3	4.3	-85.3	2.3
400	-89.6	-86.9	2.7	-89.1	0.4
500	-89.6	-89.4	0.2	-92.1	-2.5
600	-88.7	-91.4	-2.7	-94.6	-5.9
700	-92.6	-93.2	-0.6	-96.8	-4.2
800	-97.6	-94.7	2.9	-98.9	-1.3
900	-103.5	-96.2	7.4	-100.8	2.7
1000	-105.1	-97.5	7.5	-102.6	2.4
1100	-103.6	-98.8	4.8	-104.4	-0.8
1200	-104.6	-100.0	4.6	-106.2	-1.6
1300	-106.5	-101.2	5.3	-107.9	-1.4
1400	-107.5	-102.4	5.1	-109.6	-2.1
1500	-104.2	-103.6	0.6	-111.3	-7.1
1600	-104.2	-104.7	-0.5	-112.9	-8.7
1700	-108.5	-105.8	2.7	-114.5	-6.0
1800	-115.5	-106.9	8.6	-116.1	-0.6
1900	-119.1	-108.0	11.1	-117.7	1.4
2000	-118.5	-109.0	9.5	-119.3	-0.8

Magnitude of g_{22} , I_c = $2ma$, V_{ce} = $3V$					
Freq.(MHz)	Measured	SCALE	Δ %	EBBS	Δ %
100	-74.8	-75.8	-1.0	-86.0	-11.2
200	-75.5	-73.4	2.1	-82.1	-6.6
300	-72.3	-69.0	3.3	-78.3	-6.0
400	-72.6	-64.7	7.9	-74.9	-2.3
500	-71.3	-60.9	10.4	-71.7	-0.4
600	-67.7	-57.7	10.0	-68.9	-1.3
700	-64.6	-55.1	9.6	-66.5	-1.8
800	-62.5	-52.9	9.6	-64.3	-1.9
900	-60.0	-51.2	8.8	-62.5	-2.5
1000	-58.2	-49.9	8.3	-61.0	-2.7
1100	-55.8	-49.0	6.9	-59.7	-3.9
1200	-54.6	-48.2	6.4	-58.6	-4.0
1300	-56.0	-47.7	8.3	-57.7	-1.7
1400	-57.7	-47.3	10.4	-57.1	0.7
1500	-57.9	-47.1	10.8	-56.5	1.4
1600	-56.2	-47.0	9.2	-56.1	0.1
1700	-54.6	-47.0	7.6	-55.8	-1.1
1800	-54.8	-47.1	7.7	-55.6	-0.7
1900	-53.6	-47.2	6.4	-55.4	-1.8
2000	-52.8	-47.4	5.5	-55.3	-2.5

Phase of g_{22} , I_c = $2ma$, V_{ce} = $3V$					
Freq.(MHz)	Measured	SCALE	Δ deg	EBBS	Δ deg
100	84.0	82.4	-1.6	83.3	-0.8
200	82.0	75.5	-6.5	76.8	-5.2
300	69.0	69.7	0.7	70.8	1.7
400	68.0	65.0	-3.0	65.3	-2.7
500	69.0	61.4	-7.6	60.6	-8.4
600	66.0	58.7	-7.3	56.5	-9.5
700	59.0	56.7	-2.3	53.0	-6.0
800	53.0	55.3	2.3	50.0	-3.0
900	52.0	54.3	2.3	47.4	-4.6
1000	51.0	53.6	2.6	45.2	-5.8
1100	50.0	53.1	3.1	43.3	-6.7
1200	47.0	52.8	5.8	41.7	-5.3
1300	46.0	52.5	6.5	40.3	-5.7
1400	47.0	52.3	5.3	39.1	-7.9
1500	50.0	52.1	2.1	38.0	-12.0
1600	50.0	52.0	2.0	37.0	-13.0
1700	46.0	51.9	5.9	36.1	-9.9
1800	43.0	51.8	8.8	35.3	-7.7
1900	42.0	51.6	9.6	34.5	-7.5
2000	42.0	51.5	9.5	33.8	-8.2

Magnitude of q_{11} , $I_c = 5ma$, $V_{ce} = 3V$					
Freq.(MHz)	Measured	SCALE	Δ %	EBBS	Δ %
100	246.1	275.6	12.0	245.8	-0.1
200	161.7	165.6	2.4	153.4	-5.1
300	108.1	120.6	11.5	110.1	1.8
400	81.3	98.0	20.5	86.8	6.8
500	70.2	85.1	21.3	72.8	3.8
600	65.1	77.1	18.4	63.6	-2.2
700	58.3	71.7	23.0	57.3	-1.7
800	52.7	68.0	29.0	52.8	0.1
900	48.1	65.4	35.9	49.4	2.6
1000	47.0	63.4	34.9	46.8	-0.4
1100	46.0	61.9	34.6	44.8	-2.6
1200	41.9	60.8	44.9	43.2	3.1
1300	39.2	59.9	52.6	42.0	7.1
1400	38.4	59.1	54.1	41.0	6.8
1500	38.1	58.6	53.6	40.2	5.3
1600	37.5	58.1	55.1	39.5	5.3
1700	35.8	57.7	61.1	38.9	8.7
1800	35.2	57.4	63.2	38.5	9.3
1900	35.0	57.2	63.4	38.1	8.9
2000	34.7	57.0	64.2	37.7	8.8

Phase of q_{11} , $I_c = 5ma$, $V_{ce} = 3V$					
Freq.(MHz)	Measured	SCALE	Δ deg	EBBS	Δ deg
100	-46.2	-45.2	1.0	-42.3	3.8
200	-57.6	-52.4	5.2	-54.8	2.8
300	-61.7	-50.8	10.9	-57.3	4.3
400	-61.2	-47.0	14.2	-56.6	4.7
500	-58.7	-43.0	15.7	-54.6	4.1
600	-55.1	-39.1	16.0	-52.2	2.9
700	-52.7	-35.7	17.0	-49.7	3.0
800	-49.2	-32.7	16.6	-47.3	2.0
900	-47.1	-30.0	17.1	-45.0	2.1
1000	-45.0	-27.7	17.3	-42.8	2.1
1100	-42.8	-25.6	17.1	-40.9	1.9
1200	-41.3	-23.9	17.5	-39.2	2.2
1300	-39.7	-22.3	17.4	-37.6	2.1
1400	-39.1	-20.9	18.2	-36.2	2.9
1500	-38.0	-19.7	18.3	-34.9	3.0
1600	-36.8	-18.6	18.3	-33.8	3.0
1700	-36.0	-17.6	18.4	-32.8	3.2
1800	-35.1	-16.7	18.4	-32.0	3.1
1900	-34.2	-15.9	18.3	-31.2	3.0
2000	-33.7	-15.1	18.5	-30.5	3.1

Magnitude of s_{12} , I_c = $5ma$, V_{ce} = $3V$					
Freq.(MHz)	Measured	SCALE	Δ %	EBBS	Δ %
100	0.0170	0.0137	-19.4	0.0126	-26.0
200	0.0250	0.0253	1.0	0.0237	-5.1
300	0.0330	0.0341	3.3	0.0328	-0.6
400	0.0410	0.0409	-0.2	0.0399	-2.8
500	0.0460	0.0464	1.0	0.0454	-1.3
600	0.0490	0.0513	4.7	0.0498	1.7
700	0.0530	0.0558	5.2	0.0536	1.0
800	0.0580	0.0601	3.7	0.0568	-2.1
900	0.0610	0.0644	5.6	0.0598	-2.0
1000	0.0630	0.0687	9.0	0.0626	-0.7
1100	0.0660	0.0730	10.6	0.0653	-1.1
1200	0.0670	0.0773	15.4	0.0679	1.3
1300	0.0730	0.0817	11.9	0.0705	-3.4
1400	0.0770	0.0860	11.8	0.0731	-5.1
1500	0.0780	0.0905	16.0	0.0756	-3.0
1600	0.0800	0.0949	18.6	0.0782	-2.3
1700	0.0860	0.0993	15.5	0.0808	-6.1
1800	0.0920	0.1038	12.8	0.0833	-9.4
1900	0.0960	0.1082	12.7	0.0859	-10.5
2000	0.0970	0.1127	16.2	0.0885	-8.8

Phase of s_{12} , I_c = $5ma$, V_{ce} = $3V$					
Freq.(MHz)	Measured	SCALE	Δ deg	EBBS	Δ deg
100	77.0	79.4	2.4	80.1	3.1
200	75.0	70.9	-4.1	71.2	-3.8
300	62.0	65.0	3.0	63.9	1.9
400	60.0	61.3	1.3	58.2	-1.8
500	61.0	59.2	-1.8	54.0	-7.0
600	60.0	58.2	-1.8	50.9	-9.1
700	55.0	57.7	2.7	48.7	-6.3
800	51.0	57.6	6.7	47.0	-4.0
900	52.0	57.8	5.8	45.8	-6.2
1000	50.0	58.0	8.0	44.9	-5.1
1100	50.0	58.2	8.2	44.3	-5.7
1200	47.0	58.5	11.5	43.7	-3.3
1300	48.0	58.7	10.7	43.3	-4.7
1400	50.0	58.8	8.8	42.9	-7.1
1500	53.0	58.9	5.9	42.6	-10.4
1600	53.0	59.0	6.0	42.2	-10.8
1700	49.0	58.9	9.9	41.9	-7.1
1800	46.0	58.9	12.9	41.6	-4.4
1900	46.0	58.8	12.8	41.2	-4.8
2000	46.0	58.6	12.6	40.9	-5.1

Magnitude of q_{21} , I_c = 5ma , V_{ce} = 3V					
Freq.(MHz)	Measured	SCALE	Δ %	EBBS	Δ %
100	32.99	34.78	5.4	33.34	1.1
200	22.13	20.14	-9.0	20.51	-7.3
300	14.09	13.88	-1.4	14.37	2.0
400	10.46	10.54	0.8	10.98	5.0
500	8.85	8.49	-4.1	8.86	0.1
600	7.70	7.10	-7.8	7.42	-3.6
700	6.61	6.10	-7.6	6.38	-3.5
800	5.72	5.35	-6.4	5.59	-2.3
900	4.97	4.77	-4.1	4.97	0.0
1000	4.55	4.30	-5.6	4.48	-1.7
1100	4.16	3.91	-6.0	4.07	-2.3
1200	3.70	3.59	-2.9	3.73	0.8
1300	3.38	3.32	-1.7	3.44	1.8
1400	3.18	3.09	-2.8	3.20	0.5
1500	3.02	2.89	-4.2	2.98	-1.2
1600	2.86	2.72	-5.2	2.79	-2.5
1700	2.66	2.56	-3.9	2.63	-1.4
1800	2.49	2.43	-2.8	2.48	-0.6
1900	2.34	2.30	-1.7	2.35	0.1
2000	2.22	2.19	-1.2	2.23	0.3

Phase of q_{21} , I_c = 5ma , V_{ce} = 3V					
Freq.(MHz)	Measured	SCALE	Δ deg	EBBS	Δ deg
100	-53.9	-55.3	-1.4	-49.1	4.7
200	-71.0	-72.1	-1.1	-68.4	2.6
300	-82.2	-79.3	2.9	-77.4	4.9
400	-85.7	-83.5	2.2	-82.7	2.9
500	-84.7	-86.3	-1.6	-86.7	-2.0
600	-85.5	-88.6	-3.1	-89.8	-4.3
700	-90.5	-90.4	0.1	-92.4	-1.8
800	-98.3	-92.1	6.2	-94.7	3.6
900	-103.1	-93.5	9.5	-96.8	6.2
1000	-103.8	-94.9	8.8	-98.8	5.0
1100	-102.5	-96.2	6.3	-100.7	1.8
1200	-101.8	-97.4	4.4	-102.4	-0.7
1300	-104.3	-98.5	5.8	-104.2	0.2
1400	-104.9	-99.6	5.2	-105.9	-1.0
1500	-103.3	-100.7	2.6	-107.5	-4.2
1600	-102.7	-101.8	0.9	-109.1	-6.4
1700	-106.9	-102.8	4.2	-110.7	-3.8
1800	-112.4	-103.8	8.6	-112.3	0.1
1900	-116.0	-104.8	11.2	-113.8	2.2
2000	-114.7	-105.8	9.0	-115.3	-0.6

Magnitude of y_{22} , I_c = $5ma$, V_{ce} = $3V$					
Freq.(MHz)	Measured	SCALE	Δ %	EBBS	Δ %
100	609.8	592.1	-2.9	668.1	9.6
200	354.7	317.9	-10.4	344.7	-2.8
300	223.6	230.7	3.2	241.2	7.9
400	169.4	190.4	12.4	192.1	13.4
500	150.0	168.3	12.2	164.3	9.5
600	143.7	154.8	7.7	146.8	2.2
700	134.5	145.8	8.4	135.1	0.4
800	123.6	139.4	12.8	126.6	2.4
900	120.4	134.6	11.8	120.3	-0.1
1000	117.1	130.8	11.7	115.3	-1.5
1100	115.0	127.7	11.0	111.3	-3.2
1200	113.0	125.0	10.6	108.0	-4.4
1300	105.9	122.6	15.8	105.1	-0.8
1400	97.6	120.4	23.5	102.5	5.1
1500	93.7	118.5	26.4	100.2	6.9
1600	94.6	116.6	23.2	98.1	3.7
1700	95.3	114.8	20.4	96.2	0.9
1800	94.9	113.1	19.1	94.3	-0.6
1900	92.4	111.4	20.6	92.6	0.2
2000	87.9	109.8	24.9	90.9	3.4

Phase of y_{22} , I_c = $5ma$, V_{ce} = $3V$					
Freq.(MHz)	Measured	SCALE	Δ deg	EBBS	Δ deg
100	-73.6	-68.6	5.0	-82.9	-9.3
200	-68.7	-64.6	4.0	-76.1	-7.5
300	-64.4	-58.5	5.9	-70.1	-5.7
400	-63.2	-53.0	10.2	-64.8	-1.6
500	-60.0	-48.6	11.5	-60.4	-0.3
600	-55.1	-45.1	10.0	-56.7	-1.6
700	-51.4	-42.5	9.0	-53.8	-2.4
800	-48.7	-40.5	8.2	-51.5	-2.8
900	-46.4	-39.0	7.4	-49.6	-3.3
1000	-44.1	-37.9	6.1	-48.2	-4.1
1100	-42.6	-37.2	5.4	-47.2	-4.5
1200	-41.3	-36.7	4.6	-46.4	-5.1
1300	-43.6	-36.4	7.2	-45.8	-2.2
1400	-46.0	-36.3	9.7	-45.4	0.6
1500	-46.0	-36.3	9.7	-45.2	0.8
1600	-45.0	-36.4	8.5	-45.1	-0.2
1700	-43.7	-36.6	7.1	-45.1	-1.4
1800	-43.3	-36.8	6.5	-45.2	-1.9
1900	-42.6	-37.1	5.5	-45.4	-2.8
2000	-43.2	-37.4	5.7	-45.6	-2.4

7. Conclusion

A method for extracting BJT small-signal parameters from S-parameters measurements has been described. This method was implemented in the Program EBSS. Although the results are not as accurate as can be achieved with a nonlinear optimization program, we can achieve accuracy which is acceptable for many applications. Offsetting this disadvantage is the fact that the program will run 100 to 1000 times faster than an optimizing program, while avoiding the problem of nonconvergence. It is estimated that EBSS would run in about 10 seconds on Stanford's TECAP automated measurement facility, which is driven by a Hewlett-Packard 9845 microcomputer. This opens up the possibility of using S-parameters to extract small-signal parameters for various applications in which the use of a nonlinear optimizer is unwarranted. Theoretically, EBSS in conjunction with TECAP, should be capable of producing a complete set of SPICE BJT model parameters.

Although this method is not readily adapted to more complex models, it can easily be adapted to simpler ones. If, for instance, one doesn't wish to incorporate c_z and c_z in the small-signal model, then one need only set these values to zero in the program. The model for which the program EBSS was developed is sufficiently general that it might also be used as a preprocessor to a nonlinear optimization program. EBSS will supply good estimates for the values of some of the parameters in any model. This would help reduce overall run time of the optimization program, while alleviating the problems of nonconvergence and convergence at local minima.

EBSS is not without some remaining problems. The program output is somewhat more sensitive to the data than desirable. This is primarily because an explicit weighting scheme has not been applied to the data. Fitting the data to complex quantities, rather than magnitudes, should also be attempted.

Finally, the program should be rewritten for a model which accounts for the distribution of c_{tc} and c_s across r_c.

The author feels that EBSS, in its present form, is not an industrial quality program. However, it is hoped that this report has shown that the method presented here offers some significant advantages over existing methods and should be pursued further.

Bibliography

[1] E. Khalily, "Ebers-Moll Parameters Measurement Using Y-Parameters," Internal Report, Hewlett-Packard, Santa Clara Div., August 1977.

[2] Y. Ikawa, W. Eisenstadt and R. Dutton, "Modeling of High-Speed, Large-Signal Transistor Switching Transients from S-Parameters Measurements," *IEEE Trans. Electron Devices*, vol.ED-29, NO.4, pp.669-675,April 1982

[3] I. Getreu, "Modeling the Bipolar Transistor," Tektronix Inc.,1976

[4] Mathlab Group, "MACSYMA Reference Manual," MIT,1977

[5] C. Daniel and F. Wood, "Fitting Equations to Data," Wiley-Interscience,1980

[6] P. Bevington, "Data Reduction and Error Analysis for the Physical Sciences," McGraw-Hill Book Company,1969

APPENDIX A

RAW NETWORK MODEL EQUATIONS

```
delta = ((((cs*cu+cpi*cs)*cx+cpi*cs*cu)*cz+cpi*cs*cu*cx)*rb*re*rpi*zo^2
  +(((cs*cu+cpi*cs)*cx+cpi*cs*cu)*cz+cpi*cs*cu*cx)*rb*rc*re*rpi*zo)
  *s^4
+(((((cs*cu*cz+cs*cu*cx)*qm*rb+(cs*cx+cs*cu)*cz+cpi*cs*cx+cpi*cs*cu)*re
   +(((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*cz+((cs+cpi)*cu+cpi*cs)*cx)*rb)
   *rpi
   +((cs*cx+cs*cu)*cz+cs*cu*cx)*rb*re)
   *zo^2
   +(((((cs*cu*cz+cs*cu*cx)*qm*rb+(cs*cx+cs*cu)*cz+cpi*cs*cx+cpi*cs*cu)*rc
(((2*cu+2*cpi)*cx+(cs+2*cpi)*cu+cpi*cs)*cz+2*cpi*cu*cx+cpi*cs*cu)*rb)
*re
+(((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*cz+((cs+cpi)*cu+cpi*cs)*cx)*rb*rc)
*rpi
+((cs*cx+cs*cu)*cz+cs*cu*cx)*rb*rc*re)
*zo
   +(((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*cz+cpi*cu*cx+cpi*cs*cu)
*rb*rc*re*rpi)
   *s^3
  +((((cs*cx+cs*cu)*qm*re+(cu*cz+cu*cx)*qm*rb+(cx+cu+cs)*cz+(cs+cpi)*cx
            +(cs+cpi)*cu+cpi*cs)
   *rpi
   +(cs*cx+cs*cu)*re+((cx+cu+cs)*cz+(cu+cs)*cx)*rb)
   *zo^2
   +((((cs*cx+cs*cu)*qm*rc+(2*cu*cz+2*cu*cx+cs*cu)*qm*rb+(2*cx+2*cu+cs)*cz
             +2*cpi*cx+2*cpi*cu+cpi*cs)
*re
+((cu*cz+cu*cx)*qm*rb+(cx+cu+cs)*cz+(cs+cpi)*cx+(cs+cpi)*cu+cpi*cs)*rc
+((cu+cpi)*cz+(2*cu+2*cpi)*cx+(cs+cpi)*cu+cpi*cs)*rb)
*rpi
+((cs*cx+cs*cu)*rc+((2*cx+2*cu+cs)*cz+2*cu*cx+cs*cu)*rb)*re
+((cx+cu+cs)*cz+(cu+cs)*cx)*rb*rc)
*zo
   +((((cu*cz+cu*cx+cs*cu)*qm*rb+(cx+cu+cs)*cz+cpi*cx+cpi*cu+cpi*cs)*rc
+((cu+cpi)*cz+cpi*cu)*rb)
*re
+((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*rb*rc)
*rpi+((cx+cu+cs)*cz+cu*cx+cs*cu)*rb*rc*re)
   *s^2
  +(((cx+cu)*qm*rpi+cx+cu+cs)*zo^2+(((2*cx+2*cu+cs)*qm*re
                  +(cx+cu)*qm*rc+cu*qm*rb+cz+2*cx+2*cu+cs
                  +cpi)
                  *rpi
                  +(2*cx+2*cu+cs)*re+(cx+cu+cs)*rc
                  +(cz+2*cx+cu+cs)*rb)
                  *zo
                 +(((cx+cu+cs)*qm*rc+cu*qm*rb+cz+cpi)*re
                  +(cu*qm*rb+cx+cu+cs)*rc+(cu+cpi)*rb)
                  *rpi+((cx+cu+cs)*rc+(cz+cu)*rb)*re
                 +(cx+cu+cs)*rb*rc)
   *s+zo+(qm*re+1)*rpi+re+rb
```

```
s11 = (((((-cs*cu-cpi*cs)*cx-cpi*cs*cu)*cz-cpi*cs*cu*cx)*rb*re*rpi*zo^2
+(((-cs*cu-cpi*cs)*cx-cpi*cs*cu)*cz-cpi*cs*cu*cx)*rb*rc*re*rpi*zo)
*s^4
+(((((-cs*cu*cz-cs*cu*cx)*qm*rb+(-cs*cx-cs*cu)*cz-cpi*cs*cx-cpi*cs*cu)*re
 +(((-cu-cpi)*cx+(-cs-cpi)*cu-cpi*cs)*cz+((-cs-cpi)*cu-cpi*cs)*cx)*rb)
 *rpi
 +((-cs*cx-cs*cu)*cz-cs*cu*cx)*rb*re)
 *zo^2
 +(((((-cs*cu*cz-cs*cu*cx)*qm*rb+(-cs*cx-cs*cu)*cz-cpi*cs*cx-cpi*cs*cu)*rc
  +((cs*cu+cpi*cs)*cz+cpi*cs*cu)*rb)
  *re
  +(((-cu-cpi)*cx+(-cs-cpi)*cu-cpi*cs)*cz+((-cs-cpi)*cu-cpi*cs)*cx)*rb*rc)
  *rpi
  +((-cs*cx-cs*cu)*cz-cs*cu*cx)*rb*rc*re)
  *zo
 +(((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*cz+cpi*cu*cx+cpi*cs*cu)*rb*rc*re*rpi)
 *s^3
+((((-cs*cx-cs*cu)*qm*re+(-cu*cz-cu*cx)*qm*rb+(-cx-cu-cs)*cz+(-cs-cpi)*cx
            +(-cs-cpi)*cu-cpi*cs)
 *rpi
 +(-cs*cx-cs*cu)*re+((-cx-cu-cs)*cz+(-cu-cs)*cx)*rb)
 *zo^2
 +((((-cs*cx-cs*cu)*qm*rc+cs*cu*qm*rb+cs*cz+cpi*cs)*re
  +((-cu*cz-cu*cx)*qm*rb+(-cx-cu-cs)*cz+(-cs-cpi)*cx+(-cs-cpi)*cu-cpi*cs)
   *rc+((-cu-cpi)*cz+(cs+cpi)*cu+cpi*cs)*rb)
  *rpi
  +((-cs*cx-cs*cu)*rc+(cs*cz+cs*cu)*rb)*re
  +((-cx-cu-cs)*cz+(-cu-cs)*cx)*rb*rc)
  *zo
 +((((cu*cz+cu*cx+cs*cu)*qm*rb+(cx+cu+cs)*cz+cpi*cx+cpi*cu+cpi*cs)*rc
  +((cu+cpi)*cz+cpi*cu)*rb)
  *re
  +((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*rb*rc)
  *rpi+((cx+cu+cs)*cz+cu*cx+cs*cu)*rb*rc*re)
 *s^2
+(((-cx-cu)*qm*rpi-cx-cu-cs)*zo^2+((cs*qm*re+(-cx-cu)*qm*rc+cu*qm*rb-cz+cs
                    -cpi)
                  *rpi
                  +cs*re+(-cx-cu-cs)*rc+(-cz+cu+cs)*rb)
                  *zo
                 +(((cx+cu+cs)*qm*rc+cu*qm*rb+cz+cpi)*re
                  +(cu*qm*rb+cx+cu+cs)*rc+(cu+cpi)*rb)
                  *rpi+((cx+cu+cs)*rc+(cz+cu)*rb)*re
                 +(cx+cu+cs)*rb*rc)
 *s-zo+(qm*re+1)*rpi+re+rb)
/delta
```

```
s21 = (((((2*cu+2*cpi)*cx+2*cpi*cu)*cz+2*cpi*cu*cx)*rb*re*rpi*s^3
+((((2*cu*cz+2*cu*cx)*qm*rb+(2*cx+2*cu)*cz+2*cpi*cx+2*cpi*cu)*re
 +(2*cu+2*cpi)*cx*rb)
 *rpi
 +((2*cx+2*cu)*cz+2*cu*cx)*rb*re)
 *s^2+(((2*cx+2*cu)*qm*re+2*cx+2*cu)*rpi+(2*cx+2*cu)*re+2*cx*rb)*s
-2*qm*rpi)
*zo)
/delta


s12 = ((((2*cu+2*cpi)*cx+2*cpi*cu)*cz+2*cpi*cu*cx)*rb*re*rpi*zo*s^3
+((((2*cu*cz+2*cu*cx)*qm*rb+(2*cx+2*cu)*cz+2*cpi*cx+2*cpi*cu)*re
 +(2*cu+2*cpi)*cx*rb)
 *rpi
 +((2*cx+2*cu)*cz+2*cu*cx)*rb*re)
 *zo*s^2+(((2*cx+2*cu)*qm*re+2*cx+2*cu)*rpi+(2*cx+2*cu)*re+2*cx*rb)*zo*s)
/delta
```

```
s22 = (((((-cs*cu-cpi*cs)*cx-cpi*cs*cu)*cz-cpi*cs*cu*cx)*rb*re*rpi*zo^2
    +(((cs*cu+cpi*cs)*cx+cpi*cs*cu)*cz+cpi*cs*cu*cx)*rb*rc*re*rpi*zo)
    *s^4
    +(((((-cs*cu*cz-cs*cu*cx)*qm*rb+(-cs*cx-cs*cu)*cz-cpi*cs*cx-cpi*cs*cu)*re
     +(((-cu-cpi)*cx+(-cs-cpi)*cu-cpi*cs)*cz+((-cs-cpi)*cu-cpi*cs)*cx)*rb)
     *rpi
     +((-cs*cx-cs*cu)*cz-cs*cu*cx)*rb*re)
     *zo^2
     +(((((cs*cu*cz+cs*cu*cx)*qm*rb+(cs*cx+cs*cu)*cz+cpi*cs*cx+cpi*cs*cu)*rc
      +((-cs*cu-cpi*cs)*cz-cpi*cs*cu)*rb)
      *re
      +(((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*cz+((cs+cpi)*cu+cpi*cs)*cx)*rb*rc)
      *rpi
      +((cs*cx+cs*cu)*cz+cs*cu*cx)*rb*rc*re)
      *zo
     +(((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*cz+cpi*cu*cx+cpi*cs*cu)*rb*rc*re*rpi)
     *s^3
    +((((-cs*cx-cs*cu)*qm*re+(-cu*cz-cu*cx)*qm*rb+(-cx-cu-cs)*cz+(-cs-cpi)*cx
                +(-cs-cpi)*cu-cpi*cs)
     *rpi
     +(-cs*cx-cs*cu)*re+((-cx-cu-cs)*cz+(-cu-cs)*cx)*rb)
     *zo^2
     +((((cs*cx+cs*cu)*qm*rc-cs*cu*qm*rb-cs*cz-cpi*cs)*re
      +((cu*cz+cu*cx)*qm*rb+(cx+cu+cs)*cz+(cs+cpi)*cx+(cs+cpi)*cu+cpi*cs)*rc
      +((cu+cpi)*cz+(-cs-cpi)*cu-cpi*cs)*rb)
      *rpi
      +((cs*cx+cs*cu)*rc+(-cs*cz-cs*cu)*rb)*re
      +((cx+cu+cs)*cz+(cu+cs)*cx)*rb*rc)
      *zo
     +((((cu*cz+cu*cx+cs*cu)*qm*rb+(cx+cu+cs)*cz+cpi*cx+cpi*cu+cpi*cs)*rc
      +((cu+cpi)*cz+cpi*cu)*rb)
      *re
      +((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*rb*rc)
      *rpi+((cx+cu+cs)*cz+cu*cx+cs*cu)*rb*rc*re)
     *s^2
    +(((-cx-cu)*qm*rpi-cx-cu-cs)*zo^2+((-cs*qm*re+(cx+cu)*qm*rc-cu*qm*rb+cz-cs
                         +cpi)
                      *rpi
                      -cs*re+(cx+cu+cs)*rc+(cz-cu-cs)*rb)
                      *zo
                     +(((cx+cu+cs)*qm*rc+cu*qm*rb+cz+cpi)*re
                      +(cu*qm*rb+cx+cu+cs)*rc+(cu+cpi)*rb)
                      *rpi+((cx+cu+cs)*rc+(cz+cu)*rb)*re
                     +(cx+cu+cs)*rb*rc)
     *s+zo+(qm*re+1)*rpi+re+rb)
    /delta
```

APPENDIX B

SIMPLIFIED NETWORK MODE EQUATIONS

```
delta = (((cs*cu*cz+cs*cu*cx)*qm*rb*re+(((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*cz
                   +((cs+cpi)*cu+cpi*cs)*cx)
                   *rb)
  *rpi*s^3
  +((((cs*cx+cs*cu)*qm*re+(cu*cz+cu*cx)*qm*rb+(cx+cu+cs)*cz+(cs+cpi)*cx
            +(cs+cpi)*cu+cpi*cs)
   *rpi
   +((cx+cu+cs)*cz+(cu+cs)*cx)*rb)
   *s^2+((cx+cu)*qm*rpi+cx+cu+cs)*s)
  *zo^2
  +((((cs*cu*cz+cs*cu*cx)*qm*rb*rc*re
   +(((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*cz+((cs+cpi)*cu+cpi*cs)*cx)*rb*rc)
   *rpi*s^3
   +((((cs*cx+cs*cu)*qm*rc+(2*cu*cz+2*cu*cx+cs*cu)*qm*rb)*re
+((cu*cz+cu*cx)*qm*rb+(cx+cu+cs)*cz+(cs+cpi)*cx+(cs+cpi)*cu+cpi*cs)*rc
+((cu+cpi)*cz+(2*cu+2*cpi)*cx+(cs+cpi)*cu+cpi*cs)*rb)
*rpi
+((cx+cu+cs)*cz+(cu+cs)*cx)*rb*rc)
*s^2
 +(((2*cx+2*cu+cs)*qm*re+(cx+cu)*qm*rc+cu*qm*rb+cz+2*cx+2*cu+cs+cpi)*rpi
+(cx+cu+cs)*rc+(cz+2*cx+cu+cs)*rb)
*s+1)
   *zo
  +((cu*cz+cu*cx+cs*cu)*qm*rb*rc*re+((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)
                *rb*rc)
   *rpi*s^2
  +((((cx+cu+cs)*qm*rc+cu*qm*rb)*re+(cu*qm*rb+cx+cu+cs)*rc+(cu+cpi)*rb)
   *rpi
   +(cx+cu+cs)*rb*rc)
   *s+(qm*re+1)*rpi+rb
```

```
s11 = ((((-cs*cu*cz-cs*cu*cx)*qm*rb*ra+(((-cu-cpi)*cx+(-cs-cpi)*cu-cpi*cs)*cz
                    +((-cs-cpi)*cu-cpi*cs)*cx)
                    *rb)
  *rpi*zo^2
  +((-cs*cu*cz-cs*cu*cx)*qm*rb*rc*ra+(((-cu-cpi)*cx+(-cs-cpi)*cu-cpi*cs)*cz
                     +((-cs-cpi)*cu-cpi*cs)*cx)
                     *rb*rc)
   *rpi*zo)
  *s^3
  +((((-cs*cx-cs*cu)*qm*ra+(-cu*cz-cu*cx)*qm*rb+(-cx-cu-cs)*cz+(-cs-cpi)*cx
              +(-cs-cpi)*cu-cpi*cs)
   *rpi
   +((-cx-cu-cs)*cz+(-cu-cs)*cx)*rb)
   *zo^2
   +((((-cs*cx-cs*cu)*qm*rc+cs*cu*qm*rb)*ra
    +((-cu*cz-cu*cx)*qm*rb+(-cx-cu-cs)*cz+(-cs-cpi)*cx+(-cs-cpi)*cu-cpi*cs)
     *rc+((-cu-cpi)*cz+(cs+cpi)*cu+cpi*cs)*rb)
    *rpi
    +((-cx-cu-cs)*cz+(-cu-cs)*cx)*rb*rc)
    *zo
   +((cu*cz+cu*cx+cs*cu)*qm*rb*rc*ra+((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*rb*rc)
    *rpi)
   *s^2
  +(((-cx-cu)*qm*rpi-cx-cu-cs)*zo^2+((cs*qm*ra+(-cx-cu)*qm*rc+cu*qm*rb-cz+cs
                      -cpi)
                    *rpi
                    +(-cx-cu-cs)*rc+(-cz+cu+cs)*rb)
                    *zo
                   +(((cx+cu+cs)*qm*rc+cu*qm*rb)*ra
                    +(cu*qm*rb+cx+cu+cs)*rc+(cu+cpi)*rb)
                    *rpi+(cx+cu+cs)*rb*rc)
   *s-zo+(qm*ra+1)*rpi+rb)
  /delta
```

```
s21 = ((((2*cu*cz+2*cu*cx)*qm*rb*re+(2*cu+2*cpi)*cx*rb)*rpi*s^2
+(((2*cx+2*cu)*qm*re+2*cx+2*cu)*rpi+2*cx*rb)*s-2*qm*rpi)*zo)
/delta
```

```
s12 = (((2*cu*cz+2*cu*cx)*qm*rb*re+(2*cu+2*cpi)*cx*rb)*rpi*zo*s^2
+(((2*cx+2*cu)*qm*re+2*cx+2*cu)*rpi+2*cx*rb)*zo*s)
/delta
```

```
s22 = ((((-cs*cu*cz-cs*cu*cx)*qm*rb*re+(((-cu-cpi)*cx+(-cs-cpi)*cu-cpi*cs)
                    *cz+((-cs-cpi)*cu-cpi*cs)*cx)
                    *rb)
  *rpi*zo^2
  +((cs*cu*cz+cs*cu*cx)*qm*rb*rc*re+(((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*cz
                    +((cs+cpi)*cu+cpi*cs)*cx)
                    *rb*rc)
   *rpi*zo)
  *s^3
  +((((-cs*cx-cs*cu)*qm*re+(-cu*cz-cu*cx)*qm*rb+(-cx-cu-cs)*cz+(-cs-cpi)*cx
              +(-cs-cpi)*cu-cpi*cs)
   *rpi
   +((-cx-cu-cs)*cz+(-cu-cs)*cx)*rb)
   *zo^2
   +((((cs*cx+cs*cu)*qm*rc-cs*cu*qm*rb)*re
    +((cu*cz+cu*cx)*qm*rb+(cx+cu+cs)*cz+(cs+cpi)*cx+(cs+cpi)*cu+cpi*cs)*rc
    +((cu+cpi)*cz+(-cs-cpi)*cu-cpi*cs)*rb)
    *rpi
    +((cx+cu+cs)*cz+(cu+cs)*cx)*rb*rc)
    *zo
   +((cu*cz+cu*cx+cs*cu)*qm*rb*rc*re+((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*rb*rc
    *rpi)
   *s^2
  +(((-cx-cu)*qm*rpi-cx-cu-cs)*zo^2+((-cs*qm*re+(cx+cu)*qm*rc-cu*qm*rb+cz-c
                       +cpi)
                    *rpi
                    +(cx+cu+cs)*rc+(cz-cu-cs)*rb)
                    *zo
                   +(((cx+cu+cs)*qm*rc+cu*qm*rb)*re
                    +(cu*qm*rb+cx+cu+cs)*rc+(cu+cpi)*rb)
                    *rpi+(cx+cu+cs)*rb*rc)
   *s+zo+(qm*re+1)*rpi+rb)
  /delta
```

```
delta1 = rb*rc
    *((cs*cu*cz+cs*cu*cx)*qm*re+((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*cz
                   +((cs+cpi)*cu+cpi*cs)*cx)*rpi*s^3
    +((rb*((cu*cz+cu*cx)*qm*re+(cu+cpi)*cz+(cu+cpi)*cx)
     +rc*((cs*cx+cs*cu)*qm*re+(cu*cz+cu*cx)*qm*rb+(cx+cu+cs)*cz+(cs+cpi)*cx
                 +(cs+cpi)*cu+cpi*cs))*rpi
     +((cx+cu+cs)*cz+(cu+cs)*cx)*rb*rc)*s^2
    +(((cx+cu)*qm*re+(cx+cu)*qm*rc+cz+cx+cu+cpi)*rpi
     +(cx+cu+cs)*rc+(cz+cx)*rb)*s+1


q11 = (rb*rc*((cu*cz+cu*cx+cs*cu)*qm*re+(cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)
*rpi*s^2
  +((rc*((cx+cu+cs)*qm*re+cu*qm*rb+cx+cu+cs)+rb*(cu*qm*re+cu+cpi))*rpi
   +(cx+cu+cs)*rb*rc)*s
   +(qm*re+1)*rpi+rb)
  /delta1


q21 = -(rb*((cu*cz+cu*cx)*qm*re+(cu+cpi)*cx)*rpi*s^2
  -(((cx+cu)*qm*re+cx+cu)*rpi+cx*rb)*s+qm*rpi)
  /delta1

delta2 = (rb*((cs*cu*cz+cs*cu*cx)*qm*re+((cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*cz
                        +((cs+cpi)*cu+cpi*cs)*cx)*rpi*s^3
    +(((cs*cx+cs*cu)*qm*re+(cu*cz+cu*cx)*qm*rb+(cx+cu+cs)*cz+(cs+cpi)*cx
              +(cs+cpi)*cu+cpi*cs)*rpi
     +((cx+cu+cs)*cz+(cu+cs)*cx)*rb)
     *s^2+((cx+cu)*qm*rpi+cx+cu+cs)*s)*zo
    +rb*((cu*cz+cu*cx+cs*cu)*qm*re+(cu+cpi)*cx+(cs+cpi)*cu+cpi*cs)*rpi*s^2
    +(((cx+cu+cs)*qm*re+cu*qm*rb+cx+cu+cs)*rpi+(cx+cu+cs)*rb)*s

q12 = (((-cu*cz-cu*cx)*qmre+(-cu-cpi)*cx)*rb*rpi*s^2
 +(((-cx-cu)*qmre-cx-cu)*rpi-cx*rb)*s)
  /delta2

q22 = (((rb*((cu*cz+cu*cx)*qm*re+(cu+cpi)*cz+(cu+cpi)*cx)*rpi*s^2
  +(((cx+cu)*qm*re+cz+cx+cu+cpi)*rpi+(cz+cx)*rb)*s+1)*zo
  +rb*(cu*qm*re+cu+cpi)*rpi*s+(qm*re+1)*rpi+rb)
  /delta2)
  +rc
```

APPENDIX C

USER INSTRUCTIONS WITH SAMPLE INPUT

ALSO

PROGRAM LISTING : EBSS

User's Manual

EBSS reads the data from a file called "in". The first line of this file should have listed in order, re, rc, and the number of frequency points. Below this is a 5-column format for the S-parameter measurements. The top half contains s11 and s21, the bottom half s22 and s12. The first column contains the frequency in Hz for both top and bottom halves. Enter the magnitude of s11, the phase of s11 (in degrees), the magnitude of s21 and the phase of s21 in columns two through five respectively in the top half. Enter the magnitude of s22, the phase of s22, the magnitude of s12 and the phase of s12 in columns two through five respectively in the bottom half. (See sample input file on next page.)

EBSS prompts the user to enter values which determine the frequency ranges over which the fitting is done. These are frequency points called idec, jdec and kdec. Idec determines the frequency point to start the fitting of 1/p2+1/p3. Jdec determines the frequency point to end the fitting of q11, s12, cs and 1/p2+1/p3. Kdec determines the frequency point to start the fitting for cs. For instance, entering 9 for kdec would cause the program to ignore the first eight frequency points when it finds cs. The user is also asked to enter a tolerance value. This determines when the program exits out of the Subroutine Loop. For most cases, .01 should give sufficiently convergent values.

Comments

The user is left to his own devices for determining re and rc. As the model does not account for the distribution of ctc and cs across rc, the best results are obtained if only the portion of rc which contributes to the Miller effect on ctc is entered. This is generally about 1/2 to 1/3 of the dc value for rc.

The S-parameters are for a 600MHz device. Enter 15 for idec, 19 for jdec, 9 for kdec and .01 for tol.

SAMPLE INPUT FILE (put it in a file called "in")

```
.4  15   23
1e6       .974 0      1.73 178.7
1.33e6    .975 0      1.72 -179.5
1.78e6    .976 -.1    1.72 178
2.37e6    .975 -.2    1.72 177.4
3.16e6    .976 -.4    1.72 178.1
4.22e6    .975 -.4    1.72 177.4
5.62e6    .976 -.8    1.72 176.5
7.5e6     .975 -1.1   1.71 176.2
10e6      .974 -1.5   1.71 174.6
13.3e6    .974 -2.1   1.70 172.4
17.8e6    .971 -2.7   1.68 169.8
23.7e6    .968 -3.5   1.65 166.2
31.6e6    .96  -4.5   1.62 161.8
42.2e6    .949 -5.9   1.55 156
56.2e6    .936 -7.2   1.45 149.6
75e6      .915 -8.7   1.33 142.3
100e6     .888 -10.3 1.19 134.2
133.4e6 .858 -11.9 1.04 125.6
177.8e6 .818 -13.4 .876 116.3
237.1e6 .783 -13    .738 106.9
316.2e6 .744 -15.5 .569 92.9
421.7e6 .719 -16.9 .436 85.8
562.3e6 .718 -19.3 .325 80.3
1e6       1.    -.1    1.05e-4 18.6
1.33e6    1.    -.1    1.11e-4 55.3
1.78e6    1.    -.1    1.37e-4 16.7
2.37e6    1.    -.2    2.27e-4 71.3
3.16e6    1.    -.2    3.13e-4 65.9
4.22e6    1.    -.3    3.84e-4 67.
5.62e6    .999 -.4    4.91e-4 71.9
7.5e6     .999 -.5    6.42e-4 81.8
10e6      .999 -.8    7.21e-4 72.7
13.34e6    .999 -1     1.08e-3 74.2
17.78e6    .999 -1.3   1.44e-3 78.4
23.7e6    .996 -1.8   1.98e-3 81.6
31.6e6    .995 -2.4   2.49e-3 77.4
42.2e6    .99  -3.1   3.19e-3 76.6
56.2e6    .984 -3.8   3.96e-3 75.6
75e6      .977 -4.8   5.02e-3 73.2
100e6     .967 -6     6.34e-3 71.3
133.4e6 .954 -7.4   7.01e-3 71.4
177.8e6 .934 -8.9   9.96e-3 68
237.1e6 .911 -10.9 1.21e-2 67.1
316.2e6 .876 -12.7 1.39e-2 65
421.7e6 .858 -14.5 1.67e-2 75
562.3e6 .846 -16.4 1.97e-2 81.9
```

```
program EBSS

#Extracting
#Bipolar
#Small-signal parameters from
#S-parameter measurements

complex s11(30),s21(30),s12(30),s22(30)
complex q11(30),q21(30)
real m11,m21,m12,m22
real freq(30)
common freq(30),s22(30),kdec,jdec

pi=3.141592
a=pi/180.
# program assume 50 ohm characteristic impedance
zo=50.

# load data, rc, re, number of frequency points,
# and S-parameter measurements.
        open(unit=7,status='old',file='in')
        rewind(unit=7)
        read (7,*) re, rc, npts
        do i=1,npts {
        read (7,*) freq(i),m11,a11,m21,a21
        s11(i)=cmplx(m11*cos(a*a11),m11*sin(a*a11))
        s21(i)=cmplx(m21*cos(a*a21),m21*sin(a*a21))
        }
        do i=1,npts {
        read (7,*) freq(i),m22,a22,m12,a12
        s22(i)=cmplx(m22*cos(a*a22),m22*sin(a*a22))
        s12(i)=cmplx(m12*cos(a*a12),m12*sin(a*a12))
        q11(i)=zo*(1+s11(i))/(1-s11(i))
        q21(i)=s21(i)/(s11(i)-1)

        # convert hertz to angular frequency
                freq(i)=2*pi*freq(i)
                }
                close(unit=7,status='old')

# these parameters decide what frequency points are
# used for finding the various fitting coefficients.
# they provide the implicit weighting of the data.
        print *,'enter frequency point to start fitting for 1/p2+1/p3'
        read *,idec
        print *,'enter frequency point to end fitting of q11, s12,_
         cs and 1/p2+1/p3'
        read *,jdec
        print *,'enter frequency point to start fitting for cs'
        read *,kdec
        print *,'enter tolerance'
        read *,tol

# these subroutines find the fitting coefficients
        call q21fit(freq,q21,q21o,polq21,npts)
        print *,'q21o=', q21o, '   polq21=', polq21

        call q11fit(freq,q21,q21o,q11,q11o,zerq11,jdec)
        print *,'q11o=', q11o, '   zerq11=', zerq11

        call s12fit(freq,s21,s12,q21o,s12o,zers12,jdec)
        print *,'s12o=', s12o, '   zers12=', zers12

        call ppfit(freq,q21,q21o,ppq,polq21,jdec,idec)
        print *,'1/p1(1/p2 + 1/p3)=',ppq

# this subroutine extracts the small-signal parameters
        call loop(re,rc,zo,q21o,q11o,polq21,zerq11,s12o,zers12,ppq,tol)

end
```

APPENDIX D

PROGRAM LISTING : FITTING SUBROUTINES

```
subroutine q21fit(freq,q21,q21o,pole1,npts)
# this subroutine fits the inverse of the square of q21
# to P = a + b w**2 + c w**4
# a, b, and c are solved for q21o and p1(q21)

complex q21(30)
real freq(30)
double precision sy,sx2y,sx4y,sy2,sx2y2,sx4y2,sx6y2,sx8y2,a,b,c


        sy=0; sx2y=0; sx4y=0; sy2=0; sx2y2=0; sx4y2=0; sx6y2=0; sx8y2=0

        do i=1,npts{

                x = freq(i)/100e6
                y = abs(1/q21(i))**2

                sy = sy+1/y
                sx2y = sx2y+x**2/y
                sx4y = sx4y+x**4/y
                sy2 = sy2+1/y**2
                sx2y2 = sx2y2+x**2/y**2
                sx4y2 = sx4y2+x**4/y**2
                sx6y2 = sx6y2+x**6/y**2
                sx8y2 = sx8y2+x**8/y**2

        }

        a = ((sx4y2*sx8y2-sx6y2**2)*sy+sx2y2_
                *(sx4y*sx6y2-sx2y*sx8y2)_
                +sx2y*sx4y2*sx6y2-sx4y*sx4y2**2)_
                /((sx4y2*sx8y2-sx6y2**2)*sy2-sx2y2**2*sx8y2_
                +2*sx2y2*sx4y2*sx6y2-sx4y2**3)

        b = -((sx4y*sx6y2-sx2y*sx8y2)*sy2_
                +(sx2y2*sx8y2-sx4y2*sx6y2)*sy+sx2y*sx4y2**2_
                -sx2y2*sx4y*sx4y2)_
                /((sx4y2*sx8y2-sx6y2**2)*sy2-sx2y2**2*sx8y2_
                +2*sx2y2*sx4y2*sx6y2-sx4y2**3)

        c = ((sx4y*sx4y2-sx2y*sx6y2)*sy2+(sx2y2*sx6y2-sx4y2**2)_
                *sy+sx2y*sx2y2*sx4y2-sx2y2**2*sx4y)_
                /((sx4y2*sx8y2-sx6y2**2)*sy2-sx2y2**2*sx8y2_
                +2*sx2y2*sx4y2*sx6y2-sx4y2**3)

        b = b/a

        c = c/a


        q21o = sqrt(1/a)

        pole1 = -sqrt(2/(sqrt(b**2-4*c)+b))*100e6

        return
end
```

```
subroutine q11fit(freq,q21,q21o,q11,q11o,zero,jdec)
# this subroutine fits the square of q11/q21
# to P = a + b w**2
# a and b are solved for q11o and z1(q11)

complex q21(30), q11(30)
real freq(30)

        sy=0; sx2y=0; sy2=0; sx2y2=0; sx4y2=0

        do i=1,jdec{

                x = freq(i)/1e6
                y = (q21o*abs(q11(i))/abs(q21(i)))**2

                sy = sy+1/y
                sx2y = sx2y+x**2/y
                sy2 = sy2+1/y**2
                sx2y2 = sx2y2+x**2/y**2
                sx4y2 = sx4y2+x**4/y**2
        }


        a = (sx4y2*sy-sx2y*sx2y2)/(sx4y2*sy2-sx2y2**2)

        b = (sx2y*sy2-sx2y2*sy)/(sx4y2*sy2-sx2y2**2)

        q11o = sqrt(a)

        zero = -sqrt(a/b)*1e6


        return
end


subroutine s12fit(freq,s21,s12,q21o,s12o,zers12,jdec)
# this subroutine fits the square of s12/s21
# to P = a + b w**2
# a and b are solved for s12o and z1(s12)

real freq(30)
complex s21(30), s12(30)

        sy=0; sx2y=0; sy2=0; sx2y2=0; sx4y2=0

        do i=1,jdec{

                x = freq(i)/1e9
                y = abs(q21o*s12(i)/(x*s21(i)))**2

                sy = sy+1/y
                sx2y = sx2y+x**2/y
                sy2 = sy2+1/y**2
                sx2y2 = sx2y2+x**2/y**2
                sx4y2 = sx4y2+x**4/y**2
        }

        a = (sx4y2*sy-sx2y*sx2y2)/(sx4y2*sy2-sx2y2**2)

        b = (sx2y*sy2-sx2y2*sy)/(sx4y2*sy2-sx2y2**2)

        si2o = sqrt(a)/1e9

        zersi2 = -sqrt(a/b)*1e9


        return
end


subroutine ppfit(freq,q21,q21o,ppq,polq21,jdec,idec)
# this subroutine finds ppq = 1/p1(q21)*[ 1/p2(q21) + 1/p3(q21)]

real freq(30)
complex q21(30)

        sy=0

        do i=idec,jdec{

                x = freq(i)
                y = aimag(q21o/(q21(i)*cmplx(1,-freq(i)/polq21)))

                sy = sy + y/x
        }

        ppq = -1/polq21*(sy/(jdec-idec+1))


        return
end
```

APPENDIX E

PROGRAM LISTING : LOOP AND CCS SUBROUTINES

```
subroutine loop(re,rc,zo,q21o,q11o,polq21,zerq11,s12o,zers12,ppq,tol)
# this subroutine extracts the small-signal parameters
# and prints them out

real kappaf, kappap, kappaz

# save zeroq and poleq for correction factors kappaz and kappap
        zero1 = zerq11
        pole1 = polq21

# effective rc is rc+zo in all equations (except subroutine ccs)
        rc = rc+zo

# beqin loop iteratations (iterations set to 5, more than
# enough to quarentee  converqence to final values).
repeat {
        rbs = rb; rpis = rpi; qms = qm; cpis = cpi
        czs = cz; cus = cu; cxs = cx; css = cs

# block finds rb and qm
        foo = (q11o/q21o)*(zerq11-polq21)/zerq11 - kappaf
        rb = q11o-q21o*foo-re
        qm = q21o/(q11o-rb-q21o*re-re)


# block finds rpi
        rpi=q21o/qm

# block finds ctc
        ctc = (s12o+cu*rb)/q11o

# block finds cte and cpi
        cte = -(polq21*((ctc*qm*rc+ctc)*rpi+ctc*rc)+1)/(polq21*rpi)
        cpi = cte-cz

# block finds cu and cx
        a = (qm*rb*re+rb)*rpi*zers12
        b = ((cz-ctc)*qm*rb*re+(cz-cte-ctc)*rb)*rpi*zers12-rb
        c = (-ctc*qm*re-ctc)*rpi-ctc*cz*qm*rb*re*rpi*zers12
        cx = -(b-sqrt(b**2-4*a*c))/(2*a)
        cu = ctc-cx

# subroutine finds cs
        call ccs(re,rc,zo,rb,rpi,qm,cu,cx,cpi,cz,cs)

# block finds cz

        a = -rb*rpi
        b = (cu*qm*rb*re+cu*qm*rb*rc+(-cx+cu+cte)*rb)*rpi+(cx+cu+cs)*rb*rc
        c = (((cs*cx+cs*cu)*qm*rc+cu*cx*qm*rb)*re+(cu*cx*qm*rb+(cte+cs)*cx_
                +(cte+cs)*cu+cs*cte)*rc+(cu+cte)*cx*rb)*rpi+(cu+cs)*cx*rb*rc-pp
        cz = -(b-sqrt(b**2-4*a*c))/(2*a)


# block finds correction term to foo
        kappaf = -(-ctc*qm*rc**2+rc*(ctc*qm*rc-cx*qm*rb)+ctc*rc+(-cz-cx)*rb)_
        /((ctc*qm**2*rc+(cte+ctc)*qm)*rpi+ctc*qm*rc)
```

```
# block finds correction term to pi of q21
        kappap = polei*(ctc*cte*rc*rpi)
        polq21 = 1/(1/polei+kappap)

# block finds correction term to zi of q11
        kappaz = ((cu*cz+cu*cx)*qm*rb*rc*re_
                +((cu+cpi)*cx+cpi*cu)*rb*rc)*rpi*zeroi/q11o
        zerq11 = 1/(1/zeroi+kappaz)
}
until (rel(rb,rbs) < tol & rel(rpi,rpis) < tol & rel(qm,qms) < tol &_
        rel(cpi,cpis) < tol & rel(cz,czs) < tol & rel(cu,cus) < tol &_
        rel(cx,cxs) < tol & rel(cs,css) < tol)

rc = rc-zo

print *,'rb=',rb
print *,'re=',re
print *,'rpi=',rpi
print *,'cu=',cu
print *,'cx=',cx
print *,'cpi=',cpi
print *,'cz=',cz
print *,'cs=',cs
print *,'rc=',rc
print *,'qm=',qm
return
end



function rel(x,xs)

rel = abs((x-xs)/x)

return
end



subroutine ccs(re,rc,zo,rb,rpi,qm,cu,cx,cpi,cz,cs)
# this subroutine finds cs by finding the difference
# between the calculated admittance and the
# measured admittance

real freq(30)
complex s22(30),q22,s
common freq(30),s22(30),kdec,jdec

sx=0; sx2=0

do i=kdec,jdec{

s = cmplx(0,freq(i))
```

```
q22 = ((((cu*cz+cu*cx)*qm*rb*re+((cu+cpi)*cz+(cu+cpi)*cx)*rb)*rpi*s**2_
+(((cx+cu)*qm*re+cz+cx+cu+cpi)*rpi+(cz+cx)*rb)*s+1)*zo_
+(cu*qm*rb*re+(cu+cpi)*rb)*rpi*s+(qm*re+1)*rpi+rb)_
/(((((cu+cpi)*cx*s+cpi*cu*s)*cz+cpi*cu*(cx*s))*rb*rpi*s**2_
+(((cu*cz+cu*cx)*qm*rb+(cx+cu)*cz+cpi*cx+cpi*cu)*rpi+((cx+cu)*cz+cu*cx)_
*rb)*s**2+((cx+cu)*qm*rpi+cx+cu)*s)*zo_
+((cu*cz+cu*cx)*qm*rb*re+((cu+cpi)*cx+cpi*cu)*rb)*rpi*s**2_
+(((cx+cu)*qm*re+cu*qm*rb+cx+cu)*rpi+(cx+cu)*rb)*s)


         a = real(1/q22)

         b = aimag(1/q22)

         youtm = 1/abs(zo*(1+s22(i))/(1-s22(i))-rc+zo)

         x = (sqrt(youtm**2-a**2)-b)/freq(i)


sx = sx + 1/x
sx2 = sx2 + 1/x**2

}

cs = sx/sx2

return
end
```

APPENDIX F

SPICE INPUT DECKS FOR SIMULATIONS

Device measured with TECAP

```
s11 and s21
.width out=80
.options nomod numdgt=4
i1 1 0 ac -20m
rzo1 1 0 50
rzo2 2 0 50
rb 1 3 368
rc 4 2 15
re 5 0 .5
rpi 3 5 3781
cu 3 4 .08p
cx 1 4 .061p
cpi 3 5 3.93p
cz 1 5 .76p
gm 4 5 3 5 .01904
cs 4 0 1.3p
e11 10 0 1 0 2
v11 10 11 ac 1
r11 11 0 1
e21 21 0 2 0 2
r21 21 0 1
.print ac vm(11) vp(11) vm(21) vp(21)
.ac dec 8 1e6 1e9
.end

s22 and s12
.width out=80
.options nomod numdgt=4
i2 2 0 ac -20m
rzo1 1 0 50
rzo2 2 0 50
rb 1 3 368
rc 4 2 15
re 5 0 .5
rpi 3 5 3781
cu 3 4 .08p
cx 1 4 .061p
cpi 3 5 3.93p
cz 1 5 .76p
gm 4 5 3 5 .01904
cs 4 0 1.3p
e22 10 0 2 0 2
v22 10 22 ac 1
r22 22 0 1
e12 12 0 1 0 2
r12 12 0 1
.print ac vm(22) vp(22) vm(12) vp(12)
.ac dec 8 1e6 1e9
.end
```

Tektronix 7GHz device at Ic=2ma.

```
s11 and s21
.width out=80
.options nomod numdgt=4
i1 1 0 ac -20m
rzo1 1 0 50
rzo2 2 0 50
rb 1 3 64.2
rc 4 2 11
re 5 0 .562
rpi 3 5 1136
cu 3 4 .059p
cx 1 4 .175p
cpi 3 5 1.29p
cz 1 5 .31p
gm 4 5 3 5 .0673
cs 4 0 .34p
e11 10 0 1 0 2
v11 10 11 ac 1
r11 11 0 1
e21 21 0 2 0 2
r21 21 0 1
.print ac vm(11) vp(11) vm(21) vp(21)
.ac lin 20 100e6 2e9
.end

s22 and s12
.width out=80
.options nomod numdgt=4
i2 2 0 ac -20m
rzo1 1 0 50
rzo2 2 0 50
rb 1 3 64.2
rc 4 2 11
re 5 0 .562
rpi 3 5 1136
cu 3 4 .059p
cx 1 4 .175p
cpi 3 5 1.29p
cz 1 5 .31p
gm 4 5 3 5 .0673
cs 4 0 .34p
e22 10 0 2 0 2
v22 10 22 ac 1
r22 22 0 1
e12 12 0 1 0 2
r12 12 0 1
.print ac vm(22) vp(22) vm(12) vp(12)
.ac lin 20 100e6 2e9
.end
```

Tektronix 7GHz device at Ic=5ma.

```
s11 and s21
.width out=80
.options nomod numdqt=4
i1 1 0 ac -20m
rzo1 1 0 50
rzo2 2 0 50
rb 1 3 54.1
rc 4 2 11
re 5 0 .562
rpi 3 5 281
cu 3 4 .056p
cx 1 4 .185p
cpi 3 5 2.92p
cz 1 5 .35p
gm 4 5 3 5 .1764
cs 4 0 .36p
e11 10 0 1 0 2
v11 10 11 ac 1
r11 11 0 1
e21 21 0 2 0 2
r21 21 0 1
.print ac vm(11) vp(11) vm(21) vp(21)
.ac lin 20 100e6 2e9
.end
```

```
s22 and s12
.width out=80
.options nomod numdqt=4
i2 2 0 ac -20m
rzo1 1 0 50
rzo2 2 0 50
rb 1 3 54.1
rc 4 2 11
re 5 0 .562
rpi 3 5 281
cu 3 4 .056p
cx 1 4 .185p
cpi 3 5 2.92p
cz 1 5 .35p
gm 4 5 3 5 .1764
cs 4 0 .36p
e22 10 0 2 0 2
v22 10 22 ac 1
r22 22 0 1
e12 12 0 1 0 2
r12 12 0 1
.print ac vm(22) vp(22) vm(12) vp(12)
.ac lin 20 100e6 2e9
.end
```