Robust Rational Function Approximation Algorithm for Model Generation

Carlos P. Coelho [*] Joel R. Phillips [†] L. Miguel Silveira [*]

cfspc@algos.inesc.pt jrp@cadence.com lms@inesc.pt

[*] INESC / Cadence European Laboratories
Dept. of Electrical and Computer Engineering
Instituto Superior Técnico
Lisboa, 1000 Portugal

[†] Cadence Design Systems
San Jose, CA, 95134

Abstract

The problem of computing rational function approximations to tabulated frequency data is of paramount importance in the modeling arena. In this paper we present a method for generating a state space model from tabular data in the frequency domain that solves some of the numerical difficulties associated with the traditional fitting techniques used in linear least squares approximations. An extension to the MIMO case is also derived.

1 Introduction

In the design and analysis of communication, high-speed digital, and microwave electronic systems it is increasingly important to model previously neglected frequency-dependent effects that can have an important impact on the performance and functionality of a design [1, 2, 3]

For example, in modern communications systems employing advanced digital modulation schemes, such figures of merit as adjacent channel power rejection may depend critically on the frequency-dependent response of the passive circuit components. As a result circuit-level models of the passive components, in particular filters, must give very accurate representations of the actual measurements. However, in the design of microwave and RF communications circuitry, it is often difficult or impractical to obtain analytic or numerical models of many passive components, such as surface acoustic-wave filters, spiral inductors, and chip packages, that can be directly incorporated in circuit-level simulation. Instead, the passive devices are characterized by measuring the scattering (S) parameters of physical devices over the frequency range of interest. As another example, in high speed digital systems, impedance (Z) or admittance (Y) parameters describing transmission-line effects [4] are extracted from full-wave electromagnetic field solvers for use in designing clock and power distribution networks. It is sometimes possible to obtain models directly from the field solver [5], but in many cases, engineers are forced to result to a tedious trial-and-error process of generating, by hand, a lumped circuit model that is sufficiently accurate yet also compact.

Both examples demonstrate the need for an automatic, accurate and robust algorithm that can generate models for systems described by a set of tabulated data points. Although it is desirable that such models are as small as possible to simplify their manipulation during the design and analysis phases, more often than not, accuracy consideration make the use of complex models unavoidable if one is to achieve a good frequency fit in the model. Unfortunately, complex or high order models are hard to obtain, as numerical problems arise during their computation. Standard techniques for fitting a model to tabulated frequency data are usually based on determining the coefficients of a rational function approximation. This is typically done with a least squares procedure using the standard monomial basis. For high orders, such techniques lead to ill-conditioned problems and are in general not easy or even impossible to apply.

In this paper we present a method for generating a state space model from tabulated frequency-domain data that solves some of the numerical difficulties associated with the traditional fitting techniques used in linear least squares approximations. We start in Section 2 with an overview of the problem of computing rational approximations to tabulated data. Then in Section 3 we present an algorithm that robustly computes rational function approximations using the Levenberg-Marquardt method. We discuss the issues related to the numerical rank deficiency of the fitting procedures, show how to accurately and robustly compute a rational approximation, how to directly obtain an equivalent state-space representation and also how to dramatically improve the choice of a good starting point for the fitting parameters. In Section 5 we then show how the proposed techniques can be extended to compute models for multiple input, multiple output (MIMO) systems. Finally in Section 6 we present some results from the application of the method and in Section 7 some conclusions are drawn.

2 Rational approximation as a nonlinear least-squares problem

Given a set of data points describing the behavior of a system in the frequency domain, i.e. a sampled transfer function, $\boldsymbol{H}(s) = \{H(s_i)\}$, $i = 1, \cdots, N$, the rational approximation problem amounts to generating a state space model

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\,u \\ y &= \boldsymbol{C}\boldsymbol{x} + \boldsymbol{D}\,u \end{aligned} \tag{1}$$

or its equivalent rational function form

$$\boldsymbol{Y}(\boldsymbol{a}, \boldsymbol{b}, s) = \boldsymbol{C}\,(\boldsymbol{I} - s\boldsymbol{A})^{-1}\,\boldsymbol{B} + \boldsymbol{D} = \frac{\sum_{k=0}^{m} b_k s^k}{\sum_{k=0}^{n} a_k s^k} \tag{2}$$

DAC 99, New Orleans, Louisiana
(c) 1999 ACM 1-58113-109-7/99/06..$5.00

where $a_0 = 1$, $\boldsymbol{a} = [a_1, \cdots, a_n]^T$, $\boldsymbol{b} = [b_0, \cdots, b_m]^T$ and such that

$$\boldsymbol{E}(\boldsymbol{a}, \boldsymbol{b}, s) = \boldsymbol{Y}(\boldsymbol{a}, \boldsymbol{b}, s) - \boldsymbol{H}(s), \tag{3}$$

evaluated on the discrete frequency points, is minimized in some appropriate norm.

Since we are considering a discrete set of frequency points, in practice this implies the minimization of

$$\|\boldsymbol{E}(\boldsymbol{a}, \boldsymbol{b}, s)\|_2 = \sqrt{\sum_{i=1}^{N} \|\boldsymbol{Y}(\boldsymbol{a}, \boldsymbol{b}, s_i) - H(s_i)\|_2^2} \tag{4}$$

Minimization of (3) can be achieved by solving the equivalent problem of minimizing

$$\begin{aligned} f(\boldsymbol{x}) &= \frac{1}{2}\|\boldsymbol{E}(\boldsymbol{a}, \boldsymbol{b}, s)\|_2^2 = \frac{1}{2}\check{\boldsymbol{E}}(\boldsymbol{a}, \boldsymbol{b}, s)^T \check{\boldsymbol{E}}(\boldsymbol{a}, \boldsymbol{b}, s) \\ \check{\boldsymbol{E}}^T &= \left[Re\{\boldsymbol{E}\}^T \; ; \; Im\{\boldsymbol{E}\}^T \right] \end{aligned} \tag{5}$$

where $\boldsymbol{x}^T = \left[\boldsymbol{a}^T \; ; \; \boldsymbol{b}^T\right]$ and $\check{\boldsymbol{E}}$ is used instead of $\boldsymbol{E}$ in order to guarantee that the coefficients of the approximation, $\boldsymbol{a}$ and $\boldsymbol{b}$, are real numbers and thus that the model has a real time domain response. A well known algorithm for solving this problem is the Levenberg-Marquardt method, shown as Algorithm 1, which is a Gauss-Newton type method. The Levenberg-Marquardt method uses an affine model $\check{\boldsymbol{M}}$ of $\check{\boldsymbol{E}}(\boldsymbol{x})$ around $\boldsymbol{x}_c$ [6],

$$\check{\boldsymbol{M}}(\boldsymbol{x}_c) = \check{\boldsymbol{E}}(\boldsymbol{x}_c) + \boldsymbol{J}(\boldsymbol{x}_c)(\boldsymbol{x} - \boldsymbol{x}_c) \tag{6}$$

where $\boldsymbol{J}(\boldsymbol{x}_c)$ is the Jacobian of $\check{\boldsymbol{E}}$ at $\boldsymbol{x}_c$. The method minimizes, at each step, the model error, subject to a trust region type constraint

$$\begin{aligned} &\min \|\check{\boldsymbol{E}}(\boldsymbol{x}_c) + \boldsymbol{J}(\boldsymbol{x}_c)\boldsymbol{\delta x}\|_2 \\ &\qquad \text{with } \boldsymbol{\delta x} \in \mathbb{R}^{m+n+1} \text{ and } \|\boldsymbol{\delta x}\|_2 \leq \delta_c. \end{aligned} \tag{7}$$

To solve (7) we first QR factor $\boldsymbol{J} = \boldsymbol{QR}$ and determine the Gauss-Newton step $\boldsymbol{\delta x}^{GN} = -\boldsymbol{R}^{-1}\boldsymbol{Q}^T\check{\boldsymbol{E}}(\boldsymbol{x}_c)$. If this step is not satisfactory either because $\|\boldsymbol{\delta x}^{GN}\|_2 > \delta_c$ or $\|\check{\boldsymbol{E}}(\boldsymbol{x}_c)\|_2$ has not satisfied our sufficient decrease criteria, we solve the linear least squares problem

$$\min \|\boldsymbol{A\delta x} - \boldsymbol{b}\|_2 \tag{8}$$

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{R} \\ \mu_c^{\frac{1}{2}}\boldsymbol{I} \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} \boldsymbol{Q}^T\check{\boldsymbol{E}}(x_c) \\ \boldsymbol{0} \end{bmatrix} \tag{9}$$

for increasing μ_c starting with the μ_c used in the previous iteration. For high enough μ_c the update becomes aligned with the steepest descent direction $-\boldsymbol{J}(\boldsymbol{x}_c)^T\check{\boldsymbol{E}}(\boldsymbol{x}_c)$, unless we are already at a local minima. If the solution for the current μ_c is satisfactory we decrease μ_c until $\|\check{\boldsymbol{E}}(\boldsymbol{x}_c + \boldsymbol{\delta x})\|_2$ no longer satisfies our minimum decrease condition.

From (3), it is easy to see that the Jacobian matrix of $\check{\boldsymbol{E}}(\boldsymbol{x})$ is simply the Jacobian of $\boldsymbol{Y}(\boldsymbol{x})$ separated into real and imaginary parts, so that both increment vectors $\Delta\boldsymbol{a} = [\Delta a_1, \cdots, \Delta a_n]^T$ and $\Delta\boldsymbol{b} = [\Delta b_0, \cdots, \Delta b_m]^T$ are real, i.e.,

$$\begin{bmatrix} Re\left\{\frac{1}{\boldsymbol{D}_{k-1}}\boldsymbol{U}_{0:m}\right\} \vdots Re\left\{-\frac{\boldsymbol{Y}_{k-1}}{\boldsymbol{D}_{k-1}}\boldsymbol{U}_{1:n}\right\} \\ Im\left\{\frac{1}{\boldsymbol{D}_{k-1}}\boldsymbol{U}_{0:m}\right\} \vdots Im\left\{-\frac{\boldsymbol{Y}_{k-1}}{\boldsymbol{D}_{k-1}}\boldsymbol{U}_{1:n}\right\} \end{bmatrix} \begin{bmatrix} \Delta\boldsymbol{b} \\ \Delta\boldsymbol{a} \end{bmatrix} = \begin{bmatrix} Re\{\boldsymbol{Y}_{k-1} - \boldsymbol{H}\} \\ Im\{\boldsymbol{Y}_{k-1} - \boldsymbol{H}\} \end{bmatrix} \tag{10}$$

Algorithm **1** **Levenberg-Marquardt**

```
set $\boldsymbol{x}_c = \boldsymbol{x}_0$
while (not converged) {
  form $J(\boldsymbol{x}_c)$
  QR factor $J(\boldsymbol{x}_c)$
  if (Gauss-Newton step not good) {
    while ($\mu_c$ not right) {
      solve Eqn. (8)
      evaluate $\check{\boldsymbol{E}}(\boldsymbol{x}_c + \boldsymbol{\delta x})$
    }
  }
  update $\delta_c$ and the initial guess for $\mu_c$
}
```

where $\boldsymbol{U}_{0:m}$ represents the first $m+1$ columns of the Vandermonde matrix

$$\boldsymbol{U}_{0:m} = \begin{bmatrix} 1 & s_1 & s_1^2 & \cdots & s_1^m \\ 1 & s_2 & s_2^2 & \cdots & s_2^m \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & s_N & s_N^2 & \cdots & s_N^m \end{bmatrix} \tag{11}$$

and $\boldsymbol{D}_{k-1}(s)$ and $\boldsymbol{Y}_{k-1}(s)$ are diagonal matrices with the denominator and the approximation from the previous iterations as the nonzero entries.

It is easy to verify that the condition number of $\boldsymbol{U}_{0:m}$ grows very quickly with m for most problems and that $\boldsymbol{J}$ effectively loses numerical rank very rapidly, as the order increases [7]. The loss of rank implies that the Gauss-Newton procedure is likely to stall, because the candidate solution cannot be updated in directions in which the Vandermonde matrix is deficient.

3 Using Alternative Polynomial bases

3.1 Robust basis generation

In the previous section we showed that the Jacobian of the matrix used in each step of the minimization algorithm loses rank due to insufficient numerical precision. To deal with the numerical rank deficiency problem of the Vandermonde matrix we instead generate an orthonormal polynomial basis that spans the same space as the columns of the ideal Vandermonde matrix.

From (11), it is easy to see that the columns of $\boldsymbol{U}_{0:k}$ are $\left[\boldsymbol{1}, \boldsymbol{S1}, \boldsymbol{S}^2\boldsymbol{1}, \cdots, \boldsymbol{S}^k\boldsymbol{1}\right]$ where $\boldsymbol{S} = diag([s_1, s_2, \cdots, s_N]^T)$ and $\boldsymbol{1} \in \mathbb{R}^N$ is an N column vector with all entries set to 1[1]. An orthonormal basis for $colspan\{\boldsymbol{U}_{0:k}\}$ can readily be generated using an Arnoldi process - the Arnoldi process produces the factorization

$$\boldsymbol{SV}_k = \boldsymbol{V}_k\boldsymbol{H}_k + \boldsymbol{v}_{k+1}h_{k+1,k} \tag{12}$$

where $\boldsymbol{V}_k$ has orthonormal columns (i.e. $\boldsymbol{V}_k^T\boldsymbol{V}_k = \boldsymbol{I}_k$ and $\boldsymbol{I}_k$ is the identity matrix of size k), $\boldsymbol{H}_k$ is a $k \times k$ upper Hessenberg matrix (see [8, 9] for more details) and $\boldsymbol{V}_{0:k} = [\boldsymbol{V}_k, \boldsymbol{v}_{k+1}]$ is the orthonormal basis we seek.

Notice that since $\boldsymbol{S}$ is skew-symmetric so is $\boldsymbol{H}_k$. But $\boldsymbol{H}_k$ is also upper Hessenberg by definition. Therefore it has to be a tridiagonal matrix. In this case the Arnoldi process becomes a Lanczos process and the cost of generating an order k orthogonal basis is $\mathcal{O}(Nk)$.

[1] That is, the columns form a Krylov subspace $\boldsymbol{K}_{k+1}(\boldsymbol{S}, \boldsymbol{1})$ [8]

In other words this is a process for constructing a polynomial basis defined by the three term recursion

$$sP_{j-1}(s) = \sum_{i=1}^{j+1} h_{i,j} P_{i-1}(s) = \sum_{i=j-1}^{j+1} h_{i,j} P_{i-1}(s) \tag{13}$$

where $P_j(s)$ is a polynomial of order j. The polynomial basis thus defined satisfies the discrete orthogonality relation

$$\sum_{k=1}^{N} P_j(s_k) P_i(s_k) = \delta_{ij} \tag{14}$$

In order to guarantee that the model has a real time domain response, its frequency response must obey the complex conjugate symmetry condition. This condition, already discussed when (5) was introduced, guarantees $\boldsymbol{a}$ and $\boldsymbol{b}$ to be real vectors. Therefore care must be taken such that the polynomial basis we are generating also satisfies this condition. This can be achieved by requiring that the basis polynomials are orthogonal on both sides of the frequency axis. This corresponds to generating $\boldsymbol{K}_{k+1}(\tilde{\boldsymbol{S}}, \tilde{\mathbf{1}})$ where $\tilde{\boldsymbol{S}} = diag([s; -s])$ and $\tilde{\mathbf{1}}$ is now a $2N$ column vector with all entries set to 1. In practice, a similar result can be obtained by using a modified Arnoldi process where only the real projections are used, thus generating a real coefficient polynomial basis which is orthogonal when both sides of the axis are considered.

Replacing the Vandermonde matrices in (10) by the new basis matrices we get

$$\begin{bmatrix} Re\left\{\frac{1}{\boldsymbol{D}_{k-1}}\tilde{\boldsymbol{V}}_{0:m}\right\} \vdots Re\left\{-\frac{\boldsymbol{Y}_{k-1}}{\boldsymbol{D}_{k-1}}\tilde{\boldsymbol{V}}_{1:n}\right\} \\ Im\left\{\frac{1}{\boldsymbol{D}_{k-1}}\tilde{\boldsymbol{V}}_{0:m}\right\} \vdots Im\left\{-\frac{\boldsymbol{Y}_{k-1}}{\boldsymbol{D}_{k-1}}\tilde{\boldsymbol{V}}_{1:n}\right\} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{b}} \\ \tilde{\boldsymbol{a}} \end{bmatrix} = \begin{bmatrix} Re\left\{\boldsymbol{Y}_{k-1} - \boldsymbol{H}\right\} \\ Im\left\{\boldsymbol{Y}_{k-1} - \boldsymbol{H}\right\} \end{bmatrix} \tag{15}$$

where $\tilde{\boldsymbol{a}} = [\tilde{a}_1, \cdots, \tilde{a}_n]^T$, $\tilde{\boldsymbol{b}} = [\tilde{b}_0, \cdots, \tilde{b}_m]^T$ and which can be solved in the least squares sense using the QR factorization.

The numerical stability of the process allows the solution of very high order systems, a task that was previously impossible to solve in double precision using the monomial base and the Vandermonde matrices due to loss of numerical rank. By changing from the monomial base to the new polynomial base (2) becomes

$$\tilde{\boldsymbol{R}}(\tilde{\boldsymbol{a}}, \tilde{\boldsymbol{b}}, s) = \frac{\sum_{k=0}^{m} \tilde{b}_k P_k(s)}{\sum_{k=0}^{n} \tilde{a}_k P_k(s)} = \frac{\boldsymbol{Y}(s)}{\boldsymbol{U}(s)} \tag{16}$$

where the $P_j(s)$ are given by (13.)

3.2 Generating state space models

In order to obtain a state-space representation that can be used to simulate the system in the time domain, we developed a generalized controller canonical form for polynomials defined by recursion relations. To simplify, consider $m < n$; writing (16) as

$$\boldsymbol{Y}(s) = \left(\sum_{k=0}^{m} \tilde{b}_k P_k(s)\right) \left(\sum_{k=0}^{n} \tilde{a}_k P_k(s)\right)^{-1} \boldsymbol{U}(s) \tag{17}$$

and defining $\boldsymbol{\xi}(s) = \left(\sum_{k=0}^{n} \tilde{a}_k P_k(s)\right)^{-1} \boldsymbol{U}(s)$ and $F_j(s) = P_j(s)\boldsymbol{\xi}(s)$ leads to

$$\begin{aligned} \sum_{k=0}^{m} \tilde{b}_k \left(P_k(s)\boldsymbol{\xi}(s)\right) &= \sum_{k=0}^{m} \tilde{b}_k F_k(s) = \boldsymbol{Y}(s) \\ \sum_{k=0}^{n} \tilde{a}_k \left(P_k(s)\boldsymbol{\xi}(s)\right) &= \sum_{k=0}^{n} \tilde{a}_k F_k(s) = \boldsymbol{U}(s) \end{aligned} \tag{18}$$

Applying the inverse Laplace transform to (18) then leads to

$$\begin{aligned} \sum_{k=0}^{m} \tilde{b}_k f_k(s) &= y(t) \\ \sum_{k=0}^{n} \tilde{a}_k f_k(s) &= u(t) \end{aligned} \tag{19}$$

Finally, multiplying (13) by $\boldsymbol{\xi}(s)$ and applying the inverse Laplace transform, we get

$$\frac{df_{j-1}(t)}{dt} = \sum_{i=1}^{j+1} \tilde{h}_{i,j} f_{i-1}(t), \tag{20}$$

which in our particular case leads to

$$\frac{df_{j-1}(t)}{dt} = \tilde{h}_{j+1,j} f_j(t) + \tilde{h}_{j-1,j} f_{j-2}(t),$$

i.e. a three term recurrence similar the one used to define Chebyshev polynomials [10]. Noting that (19) allows us to stop the growing recursion by expressing $f_n(t)$ as a linear function of the other $f_k(t),\ k < n$ and the input, $u(t)$,

$$f_n(t) = -a_n^{-1} \sum_{k=0}^{n-1} a_k f_k(t) + a_n^{-1} u(t), \tag{21}$$

we can use the $f_j(t)$ as state space variables to construct the state space model

$$\begin{aligned} \boldsymbol{A} &= \boldsymbol{H}_n^T - a_n^{-1} h_{n+1,n} \boldsymbol{e}_n \left[a_0 \cdots a_{n-1}\right] \\ \boldsymbol{B} &= a_n^{-1} h_{n+1,n} \boldsymbol{e}_n \\ \boldsymbol{C} &= \left[b_0 \cdots b_m\, 0 \cdots 0\right] \\ D &= 0 \end{aligned} \tag{22}$$

which can then be used as the system model for time-domain simulations.

4 Accelerating nonlinear optimization

Having dealt with the issue of the inner linear system solves, there is still the matter of choosing an adequate starting point for the Levenberg-Marquardt or any other non-linear least-squares method. We dealt with this problem by using an heuristic method which we found to converge quickly to a good starting point but which demonstrated poor local convergence properties.

An alternative to the minimization of (3) is to cast the problem as a linear minimization. To linearize the nonlinear rational function fitting problem we multiply both sides of (3) by the denominator of the rational approximation, i.e.

$$\min \left\|\tilde{\boldsymbol{E}}(a, b, s)\right\|_2 = \min \left\|\left[\sum_{k=0}^{m} b_k s^k - \boldsymbol{H} \sum_{k=0}^{n} a_k s^k\right]\right\|_2 \tag{23}$$

Minimization of (23) is now equivalent to a standard linear least-squares problem whose solution is well known.

Algorithm **2**
Iteratively Scaled Linearized Rational Fitting

```
set $k = 1, \Omega_k = \mathbf{1}$
while no good guess {
  solve

  $\begin{bmatrix} Re\left\{\Omega_k \tilde{\boldsymbol{V}}_{0:m}\right\} \vdots Re\left\{-\boldsymbol{H}\Omega_k \tilde{\boldsymbol{V}}_{1:n}\right\} \\ Im\left\{\Omega_k \tilde{\boldsymbol{V}}_{0:m}\right\} \vdots Im\left\{-\boldsymbol{H}\Omega_k \tilde{\boldsymbol{V}}_{1:n}\right\} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{b}} \\ \tilde{\boldsymbol{a}} \end{bmatrix} =$
                      $\begin{bmatrix} Re\left\{\boldsymbol{H}\Omega_k \tilde{\boldsymbol{V}}_{1:n}\right\} \\ Im\left\{\boldsymbol{H}\Omega_k \tilde{\boldsymbol{V}}_{1:n}\right\} \end{bmatrix}$

  set $\Omega_{k+1} = \frac{1}{\sum_{j=0}^{n} a_j P_j(s)}$
}
```

The same problem can now be written, using our new orthogonal basis, as

$$\min \ \tilde{\boldsymbol{E}}(a,b,s) = \min \left[\sum_{k=0}^{m} b_k P_k(s) - \boldsymbol{H} \sum_{k=0}^{n} a_k P_k(s)\right] \quad (24)$$

This problem can be put in matrix form and the results described in the previous section can be directly applied to it. Doing this leads to

$$\begin{bmatrix} Re\left\{\tilde{\boldsymbol{V}}_{0:m}\right\} \vdots Re\left\{-\boldsymbol{H}\tilde{\boldsymbol{V}}_{1:n}\right\} \\ Im\left\{\tilde{\boldsymbol{V}}_{0:m}\right\} \vdots Im\left\{-\boldsymbol{H}\tilde{\boldsymbol{V}}_{1:n}\right\} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{b}} \\ \tilde{\boldsymbol{a}} \end{bmatrix} = \begin{bmatrix} Re\left\{\boldsymbol{H}\tilde{\boldsymbol{V}}_{1:n}\right\} \\ Im\left\{\boldsymbol{H}\tilde{\boldsymbol{V}}_{1:n}\right\} \end{bmatrix} \quad (25)$$

While the linear formulation above is usually simpler and easier to solve, the model that arises from the minimization of (23) is typically not the same that would result from the solution of (3) because the error is also weighted by the denominator of the approximation. This can be a very serious problem for high order or large bandwidth problems since the magnitude of the denominator of a rational function usually spans many decades.

In order to attenuate the weighting problem, we iteratively weight the equations in (25) using the values obtained for the previous denominator. The resulting algorithm, shown as Algorithm 2, tends to quickly generate an approximation which is good enough to either be used directly (under the restrictions mentioned above) or as a starting point for the Levenberg-Marquardt algorithm, (shown previously as Algorithm 1). In this last case the resulting algorithm, shown as Algorithm 3 is a hybrid method that starts by using the Iterative Scaling procedure either for some fixed number of iterations, until a desired improvement is obtained or until no improvement is obtained. Then the algorithm switches to using the Levenbeg-Marquardt procedure to refine the solution while using the output of the previous procedure as a starting point.

In Section 6 we will show through an example, that the convergence rate of this hybrid algorithm is superior to those of standard Leveberg-Marquardt and the Iterative Scaling method by itself.

Algorithm **3 Hybrid Rational Fitting**

```
until some criteria is met {
  /* could be a fixed number of iteration, iterate until
    some desired improvement is obtained or iterate until
    no improvement */

  use Iterative Scaling Linearized
    Rational Fitting on the problem
    (Algorithm 2)
  if (satisfied) return $\boldsymbol{x}_{is}$
}

set $\boldsymbol{x}_0 = \boldsymbol{x}_{is}$
run the Levenberg-Marquardt procedure
  (Algorithm 1)
```

5 Multiport models

Using matrix rational function notation [11], the concepts and algorithms described for the Single Input Single Output (SISO) case can be generalized to the Multiple Input Multiple Output (MIMO) case. We will first express the relationship between the coefficients of a matrix rational function

$$\boldsymbol{Y}(s) = \left(\sum_{k=0}^{m} \boldsymbol{B}_k P_k(s)\right) \left(\sum_{k=0}^{n} \boldsymbol{A}_k P_k(s)\right)^{-1} U(s) \quad (26)$$

and the corresponding state space model.

For a j-input, i-output system $\boldsymbol{Y}(s) \in \mathbb{C}^i$, $\boldsymbol{U}(s) \in \mathbb{C}^j$, $\boldsymbol{A}_k \in \mathbb{R}^{j \times j}$ and $\boldsymbol{B}_k \in \mathbb{R}^{i \times i}$. Defining $\boldsymbol{\xi}(s) = \left(\sum_{k=0}^{n} \boldsymbol{A}_k P_k(s)\right)^{-1} \boldsymbol{U}(s) \in \mathbb{C}^j$ and $\boldsymbol{F}_j(s) = P_j(s)\boldsymbol{\xi}(s) \in \mathbb{C}^j$ we get

$$\begin{aligned} \boldsymbol{Y}(s) &= \sum_{k=0}^{m} \boldsymbol{B}_k \boldsymbol{F}_k(s) \\ \sum_{k=0}^{n} \boldsymbol{A}_k \boldsymbol{F}_k(s) &= \boldsymbol{U}(s) \end{aligned} \quad (27)$$

As in the SISO case, we can use these equations to close the polynomial recursion and generate a state space model which leads to

$$\begin{aligned} \boldsymbol{A} &= \boldsymbol{H}_n^T \otimes \boldsymbol{I}_j - h_{n+1,n}(\boldsymbol{e}_n \otimes \boldsymbol{I}_j) \left[\boldsymbol{A}_n^{-1}\boldsymbol{A}_0 \cdots \boldsymbol{A}_n^{-1}\boldsymbol{A}_{n-1}\right] \\ \boldsymbol{B} &= h_{n+1,n}(\boldsymbol{e}_n \otimes \boldsymbol{I}_j)\boldsymbol{A}_n^{-1} \\ \boldsymbol{C} &= [\boldsymbol{B}_0 \cdot \boldsymbol{B}_m 0 \cdot 0] \\ \boldsymbol{D} &= 0 \end{aligned} \quad (28)$$

where $\otimes$ is the Kronecker product [12].

The linearized system also generalizes quite nicely to the MIMO case,

$$\boldsymbol{H}(s) = \left(\sum_{k=0}^{m} \boldsymbol{B}_k P_k(s)\right) \left(\sum_{k=0}^{n} \boldsymbol{A}_k P_k(s)\right)^{-1}. \quad (29)$$

Fixing $\boldsymbol{A}_0 = \boldsymbol{I}_j$ we get

$$\boldsymbol{H}(s)P_0(s) = \sum_{k=0}^{m} \boldsymbol{B}_k P_k(s) \left(-\boldsymbol{H}(s) \sum_{k=1}^{n} \boldsymbol{A}_k P_k(s)\right) \quad (30)$$

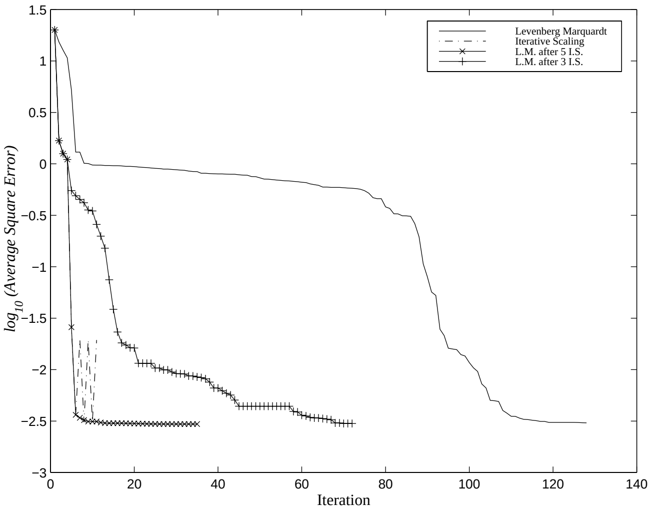

Figure 1: Convergence curves for a SAW filter example.

to solve for $\boldsymbol{A}_k$ and $\boldsymbol{B}_k$ in the least squares sense. Again, this can be represented as an overdetermined system of linear equations to be split in real and imaginary parts and solved in the least squares sense:

$$\begin{bmatrix} \boldsymbol{I}_i P_0(s_0) \cdots \boldsymbol{I}_i P_m(s_0) & \vdots & -\boldsymbol{H}_0 P_1(s_0) \cdots -\boldsymbol{H}_0 P_n(s_0) \\ \cdots \quad \cdots \quad \cdots & \vdots & \cdots \quad \cdots \quad \cdots \\ \boldsymbol{I}_i P_0(s_N) \cdots \boldsymbol{I}_i P_m(s_N) & \vdots & -\boldsymbol{H}_N P_1(s_N) \cdots -\boldsymbol{H}_N P_n(s_N) \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{B}} \\ \bar{\boldsymbol{A}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{H}_0 P_0 \\ \vdots \\ \boldsymbol{H}_N P_0 \end{bmatrix} \tag{31}$$

where $\bar{\boldsymbol{A}} = [\boldsymbol{A}_1^T, \cdots, \boldsymbol{A}_n^T]^T$, $\bar{\boldsymbol{B}} = [\boldsymbol{B}_0^T, \cdots, \boldsymbol{B}_m^T]^T$

6 Experimental Results

In this section examples will be presented that show the efficiency and robustness of the nonlinear minimization algorithm proposed. We tested the algorithm described in the previous sections by generating rational approximations to the s_{21} parameter of a SAW filter[2] and the $\boldsymbol{Y}$ parameters of a spiral inductor.

Typically a SAW filter is a narrow band filter with very high sideband rejection and a large delay. We start by comparing the convergence rates of the various algorithms when applied to an SAW filter example. Figure 1 shows a plot of the square of the approximation error with respect to the number of iterations. Shown on the plot are the modified Levenberg-Marquardt (LM) method (using the orthogonal basis scheme we described), our basic iterative scaling linearized rational fitting (IS) method and also our hybrid rational fitting algorithm (Hybrid). As can be seen from the figure the hybrid method outperforms the standard LM in speed, showing a much faster convergence, as well as the IS in robustness, thus allowing the efficient and robust generation of accurate high-order models.

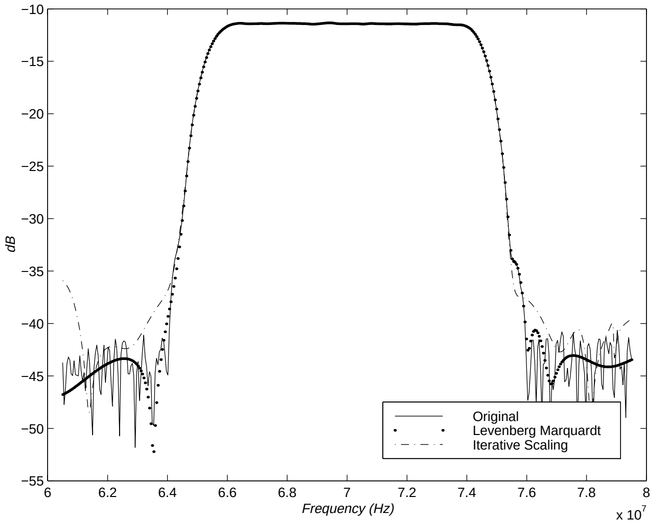

Figure 2: SAW filter s_{21} approximation using 60th order models (magnitude).

Figures 2 and 3 show sampled data and two rational approximations for, respectively, the magnitude and phase of the s_{21} parameter of a SAW filter. The rational approximations used were the modified Levenberg-Marquardt and the iterative scaling approximation. Both of the approximations shown in the figures are of order 60 and both were found to be stable. As can be seen from the figure both approximations show high accuracy in representing the original data both in the narrow band as well as in capturing the sharp transition in frequency that is typical of such systems. The rejection side bands are also well approximated even though the magnitude of both the data and the models in that region is approaching numerical noise. It would be virtually impossible to attain similar accuracy with lower-order models and the standard fitting approaches are in general unable to compute such high order models. Accurately capturing the phase by itself already requires a very high-order approximation.

For our next example we simultaneously approximated the four (4) S parameters of a spiral inductor, modeled a two-port, using the results from Section 5. In this case the frequency data spans several decades, there is a high quality resonance and some high frequency detail. Figure 4 shows plots of the magnitude of S_{11} and S_{21}. The curves shown are the original sampled data and the result of a 2×20^{th} order approximation. From the figures it is clear that the sharp zero is very accurately fitted, as well as the remaining frequency range, including the high frequency details. Again such fittings are next to impossible to obtain with low order models. The figures shows that our MIMO formulation derived in Section 5 can indeed be used to generate highly accurate models of sampled data for MIMO systems. While it is possible to approximate the system as four separate SISO systems, the resulting model would have size 4×20 which is very inefficient.

7 Conclusions and future work

In this paper we presented a robust method for generating a state space model from tabular data in the frequency domain. The method is based on a modified version of a standard nonlinear optimization algorithm. The robustness comes from a careful

[2] Data provided courtesy of John Voll, Sawtek Inc.

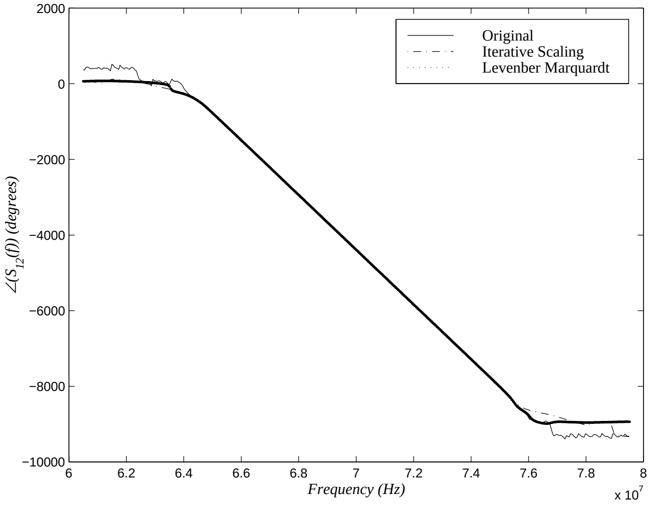

Figure 3: SAW filter s_{21} approximation using 60th order models (phase).

solution of the inner loop linear least squares problem required at each step of the nonlinear iteration. Avoiding the usage of the inherently rank deficient Vandermonde matrices that traditionally appear in such minimizations and replacing it with the computation of an orthogonal basis for the system matrix using the Arnoldi matrix, allows the generation of high order accurate models. The same technique is also used to solve a companion linear minimization that is used to obtain a good starting guess for the nonlinear minimization. We also showed that the formulation used to generate the basis can be exploited to produce a model based on a generalized controller canonical form that can be used in time-domain simulation. Results of the application of the algorithm to challenging frequency-dependent electronic systems are presented. This shows that high-order, high-accuracy state-space models can be obtained in a robust fashion.

One issue we have not addressed in this paper is the problem of generating stable and/or passive state-space models. This is a ubiquitous and pernicious problem in fitting rational functions. Fortunately, our experience with the algorithms presented in this paper is that data from passive systems often generates passive models if the frequency interval the data covers is not too narrow and the L_2 norm of the error is reasonably small. However, on occasion a few poles may fall into the wrong half-plane, making the system unstable, and in this case one of various approaches [13, 14] must be used to "stabilize" the system. To fully address the issue by providing stability or passivity guarantees is outside the scope of this paper, however, since the general case of fitting measured data is fairly broad.

References

[1] L. Miguel Silveira, Ibrahim M. Elfadel, and Jacob K. White. Efficient Frequency-Domain Modeling and Circuit Simulation of Transmission Lines. In *Proceedings of the* 31^{st} *Design Automation Conference*, pages 634–639, San Diego, CA, June 1994.

[2] Tuyen V. Nguyen, Jing Li, and Zhaojun Bai. Dispersive coupled transmission line simulation using an adaptive block lanczos algorithm. In *International Custom Integrated Circuits Conference*, pages 457–460, 1996.

[3] Guowu Zheng, Qi-Jun Zhang, Michel Nakhla, and Ramachandra

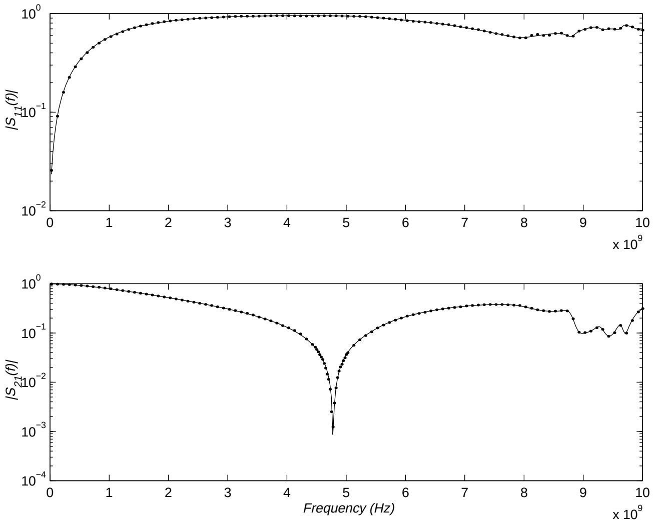

Figure 4: Magnitude of Spiral inductor Y parameter: original data (solid line) and approximation (dot markers)

Achar. An efficient approach to moment-matching simulation of linear subnetworks with measured or tabulated data. In *International Conference on Computer Aided-Design*, pages 20–23, San Jose, California, November 1996.

[4] A. Deutsch et. al. When are transmission line effects important for on-chip interconnections? *IEEE Trans. Microwave Theory and Techniques*, 45:1836–1846, October 1997.

[5] J. R. Phillips, E. Chiprout, and D. D. Ling. Efficient full-wave electromagnetic analysis via model-order reduction of fast integral transforms. In *Proceedings 33rd Design Automation Conference*, Las Vegas, Nevada, June 1996.

[6] J. E. Dennis, Jr. and Robert B. Schnabel. *Numerical Methods for Uncontrained Optimization and Nonlinear Equations*. Series in Computational Mathmatics. Prentice Hall, 1983.

[7] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Series in the Mathematical Sciences. The John Hopkins University Press, Baltimore, Maryland, third edition, 1996.

[8] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Pws Publishing Co., 1996.

[9] Lloyd N. Trefethen and David Bau. *Numerical Linear Algebra*. SIAM, 1999.

[10] Andreas Antoniou. *Digital filters analysis, design and applications*. McGraw-Hill International Editions, 2nd edition, 1993.

[11] Thomas Kailath. *Linear Systems*. Information and System Science Series. Prentice-Hall, Englewood Cliffs, New Jersey, First edition, 1980.

[12] R. S. Varga. *Matrix Iterative Analysis*. Automatic Computation Series. Prentice-Hall Inc, Englewood Cliffs, New Jersey, 1962.

[13] Zhaojun Bai, Peter Feldmann, and Roland W. Freund. Stable and passive reduced order models based on partial pade approximation via the lanczos process. Technical Report Numerical Analysis Manuscript No.97-3-10, Bell Laboratories, Lucent Technologies, Murray Hill, New Jersey, October 1997.

[14] Eli Chiprout and Michael S. Nakhla. Analysis of interconnect networks using complex frequency hopping (CFH). *IEEE Trans. CAD*, 14(2):186–200, February 1995.