Piero Triverio

8 Vector fitting

Abstract: We introduce the Vector Fitting algorithm for the creation of reduced order models from the sampled response of a linear time-invariant system. This data-driven approach to reduction is particularly useful when the system under modeling is known only through experimental measurements. The theory behind Vector Fitting is presented for single- and multiple-input systems, together with numerical details, pseudo-codes, and an open-source implementation [75]. We discuss how the reduced model can be made stable and converted to a variety of forms for use in virtually any modeling context. Finally, we survey recent extensions of the Vector Fitting algorithm geared towards time domain, parametric and distributed systems modeling.

Keywords: vector fitting, data-driven modeling, macromodeling, stability, passivity

MSC 2010: 65D10, 37M05, 93C80, 94C99

8.1 Introduction and motivation

The Vector Fitting (VF) algorithm [42, 35] is one of the most successful techniques for creating reduced order models for linear systems starting from *samples* of their response. Samples may originate from an experimental measurement or from a prior numerical simulation. This need arises in many practical scenarios, and we cite two examples.

A biomedical engineer may need a linear model describing blood flow in a portion of the human cardiovascular system, and have simultaneous in-vivo measurements of pressure and flow rate at the inlets and outlets of the region of interest. With a *data-driven* algorithm for model order reduction, such as VF, the reduced model can be created directly from experimental observations.

As a second example, we consider an electronic engineer that needs a model for a radio-frequency amplifier or an antenna, to be used for design purposes. If the device is provided by a third party, a measurement may be the only way to characterize the system. High-frequency measurements are typically performed in the frequency domain, and return the impedance or admittance seen between the ports of the device, measured at various frequencies ω_k. From these samples, VF can create a reduced model which can be represented as a set of differential equations or as an equivalent

Piero Triverio, Edward S. Rogers Sr. Department of Electrical & Computer Engineering and Institute of Biomedical Engineering, University of Toronto, 10 King's College Rd., M5S 3G4 Toronto, ON, Canada, e-mail: piero.triverio@utoronto.ca

https://doi.org/10.1515/9783110498967-008

circuit for use in subsequent simulation, including those performed in the time domain.

The main advantage of a data-driven approach to reduced order modeling is that only samples of the system response are required. This feature makes data-driven reduction a natural choice when experimental measurements are readily available. Furthermore, data-driven reduction can also be applied when samples originate from a numerical simulation based on first-principles equations, such as Maxwell's equations for electromagnetic phenomena. Although in this second scenario one could technically use *equation-driven* methods, the available simulator may not allow the user to export the discretized first-principles equations for reduction. This is the case for most commercial simulators used by industry. The main disadvantage of data-driven reduction is that it offers less physical insight into the system under modeling, since it leads to a "black-box" reduced model. By starting from a first-principles model, equation-driven methods are typically better in this regard, since they can provide to the user more information about which features of the original model were retained, and which features were discarded.

8.2 The Sanathanan–Koerner algorithm

8.2.1 Problem statement

We assume that the system under modeling is linear and time-invariant, with input $u(t) \in \mathbb{R}^{\bar{m}}$ and output $y(t) \in \mathbb{R}^{\bar{q}}$. Because of linearity and time-invariance, the output can be written as a convolution

$$y(t) = \int_{-\infty}^{+\infty} h(t-\tau)u(\tau)\,d\tau \tag{8.1}$$

between input $u(t)$ and the impulse response $h(t) \in \mathbb{R}^{\bar{q}\times\bar{m}}$ of the system, which is unknown. Applying the Laplace transform to both sides of (8.1), we get

$$Y(s) = H(s)U(s), \tag{8.2}$$

where $s = \sigma + \jmath\omega$ is complex frequency. In (8.2), $U(s) \in \mathbb{C}^{\bar{m}}$ and $Y(s) \in \mathbb{C}^{\bar{q}}$ are the Laplace transforms of $u(t)$ and $y(t)$, respectively, and $H(s) \in \mathbb{C}^{\bar{q}\times\bar{m}}$ is the transfer function of the system. The VF algorithm solves the following problem. Given $\bar{k}$ measurements of the transfer function

$$H_k = H(\jmath\omega_k) \quad k = 1, \ldots, \bar{k}, \tag{8.3}$$

determine a rational function $\widetilde{H}(s)$ that approximates the given measurements

$$\widetilde{H}(\jmath\omega_k) \simeq H_k \quad \forall k = 1, \ldots, \bar{k}. \tag{8.4}$$

In VF, $\widetilde{H}(s)$ is chosen to be a rational function. Rational functions are universal approximators, and can therefore approximate a wide range of functions with arbitrary accuracy. Moreover, since the transfer function of lumped systems is rational by construction, this is a natural choice to model dynamical systems. Finally, rational functions can be represented as a state-space system, a poles-residue form, a set of differential equations, an equivalent electric circuit and many other forms. This flexibility facilitates the integration of the reduced model into existing software for computational mathematics and system simulation.

8.2.2 The Levy and Sanathanan–Koerner algorithms

The first attempts to solve (8.4) numerically date back at least to the 1950s, with the work of Levy, Sanathanan and Koerner among others. We briefly summarize their work since the VF algorithm can be better understood from that perspective. For simplicity, we initially consider the case of a system with a single input and a single output ($\bar{m} = \bar{q} = 1$). The general case will be discussed in Section 8.3.5.

In order to solve the approximation problem (8.4), we must first choose a suitable parametric form for $\widetilde{H}(s)$, which is the model that we want to estimate from the given samples. The most natural choice is to let $\widetilde{H}(s)$ be the ratio of two polynomials

$$\widetilde{H}(s) = \frac{n(s)}{d(s)} = \frac{\sum_{n=0}^{\bar{n}} a_n s^n}{\sum_{n=0}^{\bar{n}} b_n s^n}, \tag{8.5}$$

where $a_n, b_n \in \mathbb{R}$ are unknowns, and $\bar{n}$ is the order of the desired model. Since one coefficient can be normalized, we let $b_{\bar{n}} = 1$. In (8.5), we chose the same degree $\bar{n}$ for numerator and denominator. This choice is appropriate for transfer functions that are known to be bounded when $s \to \infty$. This is the case of the scattering coefficients used to model electronic devices at high frequencies, as in the example in Section 8.3.7. In other applications, the transfer function of the system under modeling may grow polynomially as s increases. This is the case, for example, of the impedance and admittance coefficients of passive electrical circuits, which can grow linearly with s. As an example, one can consider the impedance $Z(s) = sL$ of an inductor. In such cases, the degree of the numerator of (8.5) should be increased to $\bar{n} + 1$. This change leads to minor modifications to the algorithms presented in this chapter, which will not be discussed here, but can be found in [35].

After choosing the form of model (8.5), we have to determine its coefficients a_n and b_n in order to satisfy (8.4), minimizing a suitable norm between samples H_k and model response $\widetilde{H}(\jmath\omega_k)$. We choose the l_2 norm, and aim to minimize

$$e^2 = \frac{1}{\bar{k}} \sum_{k=1}^{\bar{k}} \left| H_k - \widetilde{H}(\jmath\omega_k) \right|^2. \tag{8.6}$$

Minimizing (8.6) is a nonlinear least squares problem, due to the unknowns b_n in the denominator. Although nonlinear optimization algorithms can be directly applied to (8.6), experience shows that they can be quite time consuming and prone to local minima. A different approach is preferred, where (8.6) is linearized into a *linear* least squares problem, which can be solved efficiently and robustly with the QR decomposition [28].

We first rewrite (8.6) as

$$e^2 = \frac{1}{\bar{k}} \sum_{k=1}^{\bar{k}} \left| \frac{H_k \sum_{n=0}^{\bar{n}} b_n (\jmath\omega_k)^n - \sum_{n=0}^{\bar{n}} a_n (\jmath\omega_k)^n}{\sum_{n=0}^{\bar{n}} b_n (\jmath\omega_k)^n} \right|^2. \tag{8.7}$$

Levy proposed to linearize (8.7) by simply neglecting the denominator, and minimize [54]

$$(e_L)^2 = \frac{1}{\bar{k}} \sum_{k=1}^{\bar{k}} \left| H_k \sum_{n=0}^{\bar{n}} b_n (\jmath\omega_k)^n - \sum_{n=0}^{\bar{n}} a_n (\jmath\omega_k)^n \right|^2, \tag{8.8}$$

which ultimately boils down to solving a system of linear equations in least squares sense. Unfortunately, this simple trick typically fails to provide an accurate solution of (8.4). Indeed, error functionals (8.7) and (8.8) are equivalent only when $\sum_{n=0}^{\bar{n}} b_n (\jmath\omega)^n$ is approximately constant, which is rarely the case. Furthermore, the monomial terms $(\jmath\omega)^n$ in (8.8) will result in Vandermonde matrices in the least-squares problem to be solved, which are ill-conditioned [28].

To overcome this issue, Sanathanan and Koerner proposed an iterative process to improve the quality of the solution [69]. In the first iteration ($i = 1$), the Levy functional (8.8) is minimized, providing a first estimate of the model coefficients that we denote as $a_n^{(1)}$ and $b_n^{(1)}$. In successive iterations ($i \geq 2$), the following linearization of (8.7) is minimized:

$$\left(e_{SK}^{(i)}\right)^2 = \frac{1}{\bar{k}} \sum_{k=1}^{\bar{k}} \left| \frac{H_k \sum_{n=0}^{\bar{n}} b_n^{(i)} (\jmath\omega_k)^n - \sum_{n=0}^{\bar{n}} a_n^{(i)} (\jmath\omega_k)^n}{\sum_{n=0}^{\bar{n}} b_n^{(i-1)} (\jmath\omega_k)^n} \right|^2, \tag{8.9}$$

leading to a new estimate of model coefficients $a_n^{(i)}$ and $b_n^{(i)}$. We can see that, in (8.9), the coefficients $b_n^{(i-1)}$ from the previous iteration are used to approximate the "nonlinear" term in (8.7). Since the unknowns $a_n^{(i)}$ and $b_n^{(i)}$ appear only in the numerator, the Sanathanan–Koerner method only requires the solution of linear least squares problems. If the iterative process converges, $b_n^{(i-1)} \to b_n^{(i)}$, and (8.9) becomes equivalent to (8.7). We can see that the term $\sum_{n=0}^{\bar{n}} b_n^{(i-1)} (\jmath\omega_k)^n$ in the denominator of (8.9) acts as a frequency-dependent weight of the least squares problem. This weight aims to progressively remove the bias introduced in the linearization of (8.6). For discrete-time systems, the counterpart of the Sanathanan–Koerner method was proposed by Steiglitz and McBride [73].

8.2.3 Numerical issues of the Sanathanan–Koerner method

The work of Sanathanan and Koerner solves (8.4) accurately using only linear least squares problems. Unfortunately, this method can still suffer from severe numerical issues when applied to realistic problems, where the required model order $\bar{n}$ may be quite large and frequency ω may span several orders of magnitude. For example, VF is extensively used in integrated circuit design to model the interconnect network that distributes signals and power across the circuit. In this application, the frequency range of interest typically extends from a few MHz to tens of GHz, for about four decades of variation. The numerical issues associated with the Sanathanan–Koerner method arise from two factors:

(a) The error (8.9) contains high powers of frequency $(\omega_k)^n$, leading to very poor conditioning. Specifically, the matrix of the least-squares problem to be solved will contain Vandermonde blocks [28], which are known to be ill-conditioned even for relatively modest values of $\bar{n}$.
(b) The weighting term $\sum_{n=0}^{\bar{n}} b_n^{(i-1)}(\jmath\omega_k)^n$ in the denominator of (8.9) typically exhibits large variations over $[\omega_1, \omega_{\bar{k}}]$, which further degrade the conditioning of the least squares problem.

8.3 The Vector Fitting algorithm

The VF algorithm, conceived by Gustavsen and Semlyen [42], addresses both problems with a simple yet brilliant solution.

8.3.1 A new basis function and implicit weighting

In order to avoid the ill-conditioning arising from the s^n terms in (8.5), VF replaces those terms with partial fractions. The numerator and denominator of $\widetilde{H}(s)$ are written as

$$n^{(i)} = c_0^{(i)} + \sum_{n=1}^{\bar{n}} \frac{c_n^{(i)}}{s - p_n^{(0)}}, \tag{8.10}$$

$$d^{(i)} = 1 + \sum_{n=1}^{\bar{n}} \frac{d_n^{(i)}}{s - p_n^{(0)}}, \tag{8.11}$$

where $p_n^{(0)} \in \mathbb{C}$ are a set of initial poles, whose choice will be discussed later on. We see that, without loss of generality, the constant term in (8.11) has been normalized to one. In comparison to the monomial basis functions s^n used by the Sanathanan–Koerner iteration, which vary wildly as s increases, partial fractions $\frac{1}{s-p_n^{(0)}}$ have more contained variations over frequency if the poles $p_n^{(0)}$ are chosen appropriately [43], as

will be discussed in Section 8.3.2. This feature leads to better conditioning, especially if the poles $p_n^{(0)}$ are distinct and well separated.

The introduction of partial fractions is also crucial to address the second issue discussed in Section 8.2.3, and perform an implicit weighting of (8.9). To understand how VF achieves this, we first give a different interpretation to linearized error (8.9). In terms of (8.10) and (8.11), error (8.9) can be expressed as

$$(e_{SK}^{(i)})^2 = \frac{1}{\bar{k}} \sum_{k=1}^{\bar{k}} \left| H_k \frac{d^{(i)}(\jmath\omega_k)}{d^{(i-1)}(\jmath\omega_k)} - \frac{n^{(i)}(\jmath\omega_k)}{d^{(i-1)}(\jmath\omega_k)} \right|^2. \tag{8.12}$$

We can see that this expression involves the two new quantities

$$w^{(i)}(s) = \frac{d^{(i)}(s)}{d^{(i-1)}(s)} \tag{8.13}$$

and

$$\widetilde{H}^{(i)}(s) = \frac{n^{(i)}(s)}{d^{(i-1)}(s)}. \tag{8.14}$$

The function $\widetilde{H}^{(i)}(s)$ can be interpreted as the model transfer function estimated at iteration i by the minimization of (8.12). Notably, this transfer function is made by the numerator $n^{(i)}(s)$ from the current iteration (to be found), and by the denominator $d^{(i-1)}(s)$ from the previous iteration (already known). This approximation arises from the linearization of the error function, since it indeed avoids the presence of unknowns in the denominator. The function $w^{(i)}(s)$ can be interpreted as a frequency-dependent weight which multiplies the given samples H_k. This weighting function has two purposes:

- providing a new estimate of denominator $d^{(i)}(s)$, and
- compensating for the approximation introduced by fixing the denominator of $\widetilde{H}^{(i)}(s)$ to the previous iteration value. Indeed, weight $w^{(i)}(s)$ depends on the ratio between new denominator estimate $d^{(i)}(s)$ and previous estimate $d^{(i-1)}(s)$.

Next, we derive alternative expressions for $w^{(i)}(s)$ and $\widetilde{H}^{(i)}(s)$, which pave the way for an implicit weighting of (8.12). Substituting (8.10) and (8.11) into (8.13) we can derive the following chain of equalities:

$$w^{(i)}(s) = \frac{1 + \sum_{n=1}^{\bar{n}} \frac{d_n^{(i)}}{s-p_n^{(0)}}}{1 + \sum_{n=1}^{\bar{n}} \frac{d_n^{(i-1)}}{s-p_n^{(0)}}} = \frac{\frac{\prod(s-p_n^{(i)})}{\prod(s-p_n^{(0)})}}{\frac{\prod(s-p_n^{(i-1)})}{\prod(s-p_n^{(0)})}} = 1 + \sum_{n=1}^{\bar{n}} \frac{w_n^{(i)}}{s-p_n^{(i-1)}}, \tag{8.15}$$

where $\prod = \prod_{n=1}^{\bar{n}}$. In (8.15), $p_n^{(i)}$ are the *zeros* of $d^{(i)}(s)$, and therefore the *poles* of $\widetilde{H}^{(i+1)}(s)$. From the second expression in (8.15), we see that $w^{(i)}(s)$ is the ratio of two rational functions with the same poles $p_n^{(0)}$. By factorizing their respective numerators and denominators, as in the third expression, we observe that those common poles

can be eliminated. Finally, we express $w^{(i)}$ in terms of a new set of poles $p_n^{(i-1)}$ that change at every iteration, as in the last expression in (8.15). The same manipulation can be performed on $\widetilde{H}^{(i)}(s)$, leading to

$$\widetilde{H}^{(i)}(s) = \frac{c_0^{(i)} + \sum_{n=1}^{\bar{n}} \frac{c_n^{(i)}}{s-p_n^{(0)}}}{1 + \sum_{n=1}^{\bar{n}} \frac{d_n^{(i-1)}}{s-p_n^{(0)}}} = r_0^{(i)} + \sum_{n=1}^{\bar{n}} \frac{r_n^{(i)}}{s - p_n^{(i-1)}}. \tag{8.16}$$

Substituting (8.15) and (8.16) into (8.12), we obtain [42, 35]

$$\left(e_{SK}^{(i)}\right)^2 = \frac{1}{\bar{k}} \sum_{k=1}^{\bar{k}} \left| H_k \left(1 + \sum_{n=1}^{\bar{n}} \frac{w_n^{(i)}}{\jmath\omega_k - p_n^{(i-1)}} \right) - \left(r_0^{(i)} + \sum_{n=1}^{\bar{n}} \frac{r_n^{(i)}}{\jmath\omega_k - p_n^{(i-1)}} \right) \right|^2, \tag{8.17}$$

which is the actual error function used in VF to fit the model to the given samples. The main difference between (8.17) and (8.9) is how the linearized error is iteratively weighted to progressively converge to (8.6). In (8.12), the weight $\frac{1}{d^{(i-1)}(\jmath\omega_k)}$ is applied *explicitly*, which degrades numerical conditioning. In (8.17), instead, the same weight is applied *implicitly* by relocating the poles $p_n^{(i-1)}$ at each iteration.

Once (8.17) has been minimized, the updated poles $p_n^{(i)}$ for the next iteration can be found as the zeros of $d^{(i)}(s)$, as one can see from the third expression in (8.15). It can be shown that such zeros can be calculated as the eigenvalues of a matrix [42]

$$\{p_n^{(i)}\} = \mathrm{eig}(A^{(i-1)} - b_w (c_w^{(i)})^T), \tag{8.18}$$

with $A^{(i-1)} = \mathrm{diag}\{p_1^{(i-1)}, \ldots, p_{\bar{n}}^{(i-1)}\}$ being a diagonal matrix formed by poles $p_n^{(i-1)}$. In (8.18) b_w is a $\bar{n} \times 1$ vector of ones, and $(c_w^{(i)})^T = [w_1^{(i)}, \ldots, w_{\bar{n}}^{(i)}]$. Upon convergence, $p_n^{(i-1)} \to p_n^{(i)}$, and they become the poles of the obtained model $H^{(i)}(s)$. When this happens, $w^{(i)} \to 1$ as we can see from the third expression in (8.15), and the linearized error (8.12) tends to (8.6), as desired.

8.3.2 The Vector Fitting algorithm

We are now ready to present the complete VF algorithm [42, 35], with a pseudo-code implementation available in Algorithm 8.1. The first step is to choose the order $\bar{n}$ of the desired model. This choice will be discussed in Section 8.3.11.1. Next, we set the initial poles $p_n^{(0)}$ of the basis functions in (8.16) and (8.15). Numerical tests [42] showed that a linear distribution of poles with small and negative real part over the bandwidth spanned by samples H_k leads to the best conditioning of the least squares problems to be solved. We assume $\bar{n}$ even, and frequency values ω_k sorted in ascending order. If $\omega_1 = 0$, the initial poles can be set as [35]

$$p_n^{(0)} = \begin{cases} (-\alpha + \jmath)\frac{\omega_k}{\bar{n}/2} n & \text{for } n = 1, \ldots, \bar{n}/2 \\ (p_{n-\bar{n}/2}^{(0)})^* & \text{for } n = \bar{n}/2 + 1, \ldots, \bar{n} \end{cases} \tag{8.19}$$

Algorithm 8.1: Vector Fitting.

Require: response samples H_k, corresponding frequencies ω_k $(k = 1, \dots, \bar{k})$
Require: desired model order $\bar{n}$
Require: maximum number of iterations $i_{\max}$
set initial poles $p_n^{(0)}$ according to (8.19) or (8.20).
$i \leftarrow 1$
while $i \le i_{\max}$ **do**
    Solve (8.21) or (8.38) in least squares sense
    Compute the new poles estimate $p_n^{(i)}$ with (8.18)
    Enforce poles stability with (8.71), if desired ▷ Stability enforcement
    if (8.27) is true **then** ▷ First convergence test
        Solve (8.29) or (8.40) in least squares sense ▷ Tentative final fitting
        Compute fitting error e with (8.6) or (8.34)
        if $e \le \varepsilon_H$ **then** ▷ Second convergence test
            $\widetilde{H}(s) = \widetilde{H}^{(i+1)}(s)$
            return Success!
        end if
    end if
    $i \leftarrow i + 1$
end while
return Failure: maximum number of iterations reached.

where * denotes the complex conjugate and α is typically set to 0.01. This rule generates $\bar{n}/2$ pairs of complex conjugate poles, linearly distributed over the frequency range $[0, \omega_{\bar{k}}]$ spanned by samples H_k. The imaginary part of the poles is set to be quite larger than the real part, since this makes the partial fraction basis functions well distinct from each other, which improves numerical conditioning.

When $\omega_1 \neq 0$, the distribution (8.19) can be modified as [35]

$$p_n^{(0)} = \begin{cases} (-\alpha + \jmath)[\omega_1 + \frac{\omega_{\bar{k}} - \omega_1}{\bar{n}/2-1}(n-1)] & \text{for } n = 1, \dots, \bar{n}/2, \\ (p_{n-\bar{n}/2}^{(0)})^* & \text{for } n = \bar{n}/2+1, \dots, \bar{n}, \end{cases} \tag{8.20}$$

to linearly spread the poles between $\omega = \omega_1$ and $\omega = \omega_{\bar{k}}$. Rules (8.19) and (8.20) work well for most cases, since the choice of initial poles is typically not critical for VF convergence. When the frequency range of interest spans several decades, and the system frequency response exhibits significant behavior in multiple decades, initial poles can be distributed logarithmically for optimal results [35].

The core of the VF algorithm is an iterative minimization of (8.17), which begins with $i = 1$. Minimizing (8.17) is equivalent to solving, in the least-squares sense, the

system of equations

$$\begin{bmatrix} \Phi_0^{(i)} & -D_H \Phi_1^{(i)} \end{bmatrix} \begin{bmatrix} c_H^{(i)} \\ c_w^{(i)} \end{bmatrix} = V_H \tag{8.21}$$

where $\Phi_0^{(i)}$ and $\Phi_1^{(i)}$ contain the partial fraction basis functions evaluated at the different frequency points ω_k:

$$\Phi_0^{(i)} = \begin{bmatrix} 1 & \frac{1}{\jmath\omega_1 - p_1^{(i-1)}} & \cdots & \frac{1}{\jmath\omega_1 - p_{\bar{n}}^{(i-1)}} \\ \vdots & \vdots & & \vdots \\ 1 & \frac{1}{\jmath\omega_{\bar{k}} - p_1^{(i-1)}} & \cdots & \frac{1}{\jmath\omega_{\bar{k}} - p_{\bar{n}}^{(i-1)}} \end{bmatrix}, \tag{8.22}$$

$$\Phi_1^{(i)} = \begin{bmatrix} \frac{1}{\jmath\omega_1 - p_1^{(i-1)}} & \cdots & \frac{1}{\jmath\omega_1 - p_{\bar{n}}^{(i-1)}} \\ \vdots & & \vdots \\ \frac{1}{\jmath\omega_{\bar{k}} - p_1^{(i-1)}} & \cdots & \frac{1}{\jmath\omega_{\bar{k}} - p_{\bar{n}}^{(i-1)}} \end{bmatrix}, \tag{8.23}$$

and $D_H = \mathrm{diag}\{H_1, \ldots, H_{\bar{k}}\}$. The right hand side of (8.21) is a column vector formed by the given samples

$$V_H = \begin{bmatrix} H_1 & \ldots & H_{\bar{k}} \end{bmatrix}^T, \tag{8.24}$$

while $c_H^{(i)}$ and $c_w^{(i)}$ contain the unknown coefficients

$$c_H^{(i)} = \begin{bmatrix} r_0^{(i)} & \ldots & r_{\bar{n}}^{(i)} \end{bmatrix}^T, \tag{8.25}$$

$$c_w^{(i)} = \begin{bmatrix} w_1^{(i)} & \ldots & w_{\bar{n}}^{(i)} \end{bmatrix}^T. \tag{8.26}$$

System (8.21) can be solved in the least-squares sense with a QR decomposition of the coefficient matrix [28]. Once (8.21) has been solved, the new poles estimate $p_n^{(i)}$ is computed with (8.18).

The VF iterative process usually converges very quickly, often in 4–5 iterations, except when the given samples are noisy. The fast and reliable convergence of VF is truly remarkable considering that VF ultimately solves a nonlinear minimization problem. Unfortunately, so far no one has been able to support this experimental evidence with strong theoretical results on VF convergence. Actually, contrived examples show that VF convergence is not guaranteed [53, 72]. However, these examples are quite artificial and far from practical datasets. Two decades of widespread use indeed show that, when properly implemented, VF is a remarkably robust algorithm for the identification of reduced order models from sampled data. In VF, convergence is typically monitored with three conditions:

1. When the poles estimates stabilizes, i. e. $p_n^{(i)} \simeq p_n^{(i-1)}$, performing new iterations will not improve accuracy. When this happens, $w^{(i)}(\jmath\omega) \simeq 1$ for $\omega \in [\omega_1, \omega_{\bar{k}}]$. This occurrence can be tested numerically as

$$\sqrt{\frac{1}{\bar{k}} \sum_{k=1}^{\bar{k}} \left| w^{(i)}(\jmath\omega) - 1 \right|^2} \leq \varepsilon_w, \tag{8.27}$$

where ε_w is a user-defined threshold. The advantage of criterion (8.27) is that it does not require additional computations apart from the calculation of the norm in (8.27). The limitation of this test is that it is based on an empirical condition, which may be satisfied even when the reduced model $H^{(i)}(s)$ still does not fit well the given frequency samples. Conversely, when samples H_k are noisy, test (8.27) may fail even when additional iterations will not significantly improve the poles estimate [33]. Therefore, test (8.27) should be only used as a low-cost check to decide whether it is worth to compute the error between the reduced model response and samples H_k.
2. When condition (8.27) is satisfied, the error between the fitted model and samples H_k should be checked. In principle, this can be done by computing the error between (8.16) and H_k. However, since after solving (8.21) a new estimate of the poles can be found via (8.18), the common practice is to use those poles to fit a new model. This is done by minimizing the exact error (8.6) between the given samples H_k and model

$$\widetilde{H}^{(i+1)}(s) = r_0^{(i+1)} + \sum_{n=1}^{\bar{n}} \frac{r_n^{(i+1)}}{s - p_n^{(i)}}, \tag{8.28}$$

considering only residues $r_0^{(i+1)}, \ldots, r_{\bar{n}}^{(i+1)}$ as unknowns. Since poles $p_n^{(i)}$ are now fixed, this is equivalent to solve, in least squares sense, the linear system

$$\Phi_0^{(i+1)} c_H^{(i+1)} = V_H. \tag{8.29}$$

The VF iteration ends, successfully, when

$$e \leq \varepsilon_H, \tag{8.30}$$

since model (8.28) meets the accuracy threshold ε_H set by the user. The main reason why this additional fitting step is performed is because this step minimizes the exact error (8.6) between model and given samples, rather than a linear approximation like (8.17), which improves accuracy and more reliably detects convergence. Therefore, solving (8.29) serves both as convergence test and as final fitting of the model.
3. In selected circumstances, VF may be unable to reach (8.30) even after many iterations. In this case, the iterative process concludes unsuccessfully when i exceeds the maximum number of iterations $i_{\max}$ allowed by the user.

8.3.3 Example: fitting a rational transfer function

In this example, we apply VF to a set of samples H_k generated from a known rational function of order 10. Its poles were generated randomly, and are reported in Table 8.1. The original transfer function was sampled at $\bar{k} = 100$ frequency points linearly spaced between $\omega_1 = 0.1$ rad/s and $\omega_{100} = 10$ rad/s. A Matlab implementation of VF [75] was used to fit the samples with a model in the form (8.16) with order $\bar{n} = 10$. The initial distribution of poles $p_n^{(0)}$ set by (8.20) is depicted in the left panel of Figure 8.1. Throughout the VF iterations, poles relocate to the final distribution shown in the right panel of Figure 8.1, which also compares them to the exact poles of the original rational function. We can see that the poles estimated by VF closely match the poles of the original system.

Table 8.1: Example of Section 8.3.3: poles and residues of the transfer function used to generate samples H_k.

Pole	Residue
constant term	$r_0 = 0.1059$
$p_1 = -1.3578$	$r_1 = -0.2808$
$p_2 = -1.2679$	$r_2 = 0.1166$
$p_{3,4} = -1.4851 \pm 0.2443\jmath$	$r_{3,4} = 0.9569 \mp 0.7639\jmath$
$p_{5,6} = -0.8487 \pm 2.9019\jmath$	$r_{5,6} = 0.9357 \mp 0.7593\jmath$
$p_{7,8} = -0.8587 \pm 3.1752\jmath$	$r_{7,8} = 0.4579 \mp 0.7406\jmath$
$p_{9,10} = -0.2497 \pm 6.5369\jmath$	$r_{9,10} = 0.2405 \mp 0.7437\jmath$

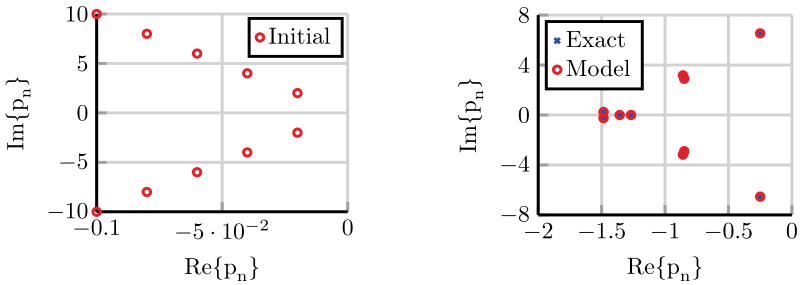

Figure 8.1: Left panel: initial poles $p_n^{(0)}$ used by VF in the first iteration. Right panel: poles of the final model $\widetilde{H}(s)$ compared to the exact poles of the original transfer function.

In Figure 8.2, the frequency response $\widetilde{H}(\jmath\omega)$ of the VF model is compared to the initial samples. We observe excellent agreement over the entire frequency range of interest. At the conclusion of the VF iterative process, the worst-case error between samples H_k and model response

$$e_\infty = \max_k |H_k - H(\jmath\omega_k)| \tag{8.31}$$

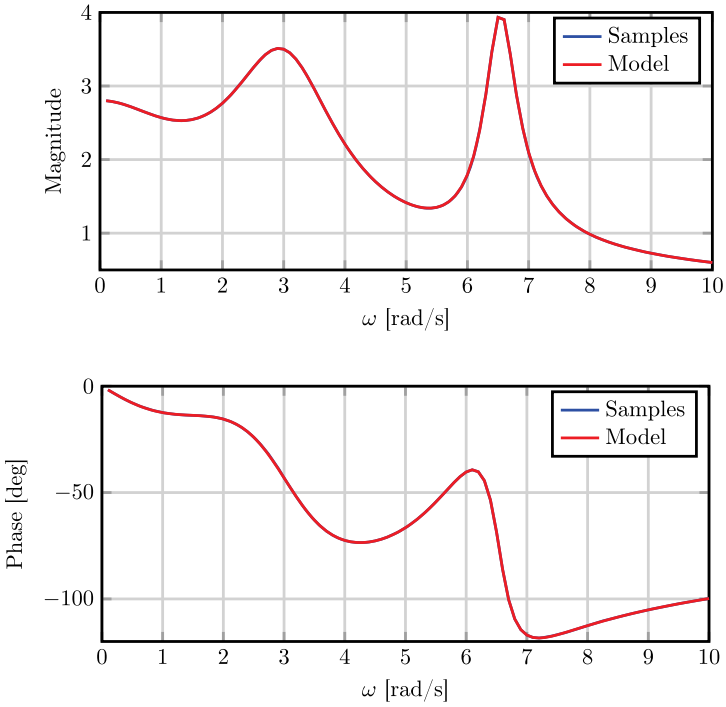

Figure 8.2: Example of Section 8.3.3: magnitude (top) and phase (bottom) of samples H_k and of the model $\tilde{H}(\jmath\omega)$ identified by VF.

is 2.37×10^{-14}. Figure 8.3 shows the evolution of e_∞ throughout the five iterations performed by VF, plus a final iteration ($i = 6$) where poles were kept fixed and residues were calculated one more time using (8.29). The figure shows that VF converges very quickly, reaching an error below 10^{-8} in only three iterations. We can also observe that the final fitting iteration ($i = 6$) with fixed poles provides a more accurate model. For this example, VF took only 0.2 s of CPU time on a 2.2 GHz mobile processor. The source codes related to this example can be downloaded from [75].

8.3.4 Example: modeling of aortic input impedance

In this example, VF is used to model the relation between pressure $p(t)$ and flow rate $q(t)$ in the ascending aorta of a 1.1-year old patient [71, patient 1]. Simultaneous pressure and flow rate measurements were collected during a surgical procedure. The blood flow rate was measured with an ultrasonic flow probe positioned about 1 cm downstream of the aortic valve. The pressure was acquired using a catheter with a pressure transducer on its tip, positioned in the same location as the flow rate probe. From time domain recordings, the input impedance seen from the aorta was

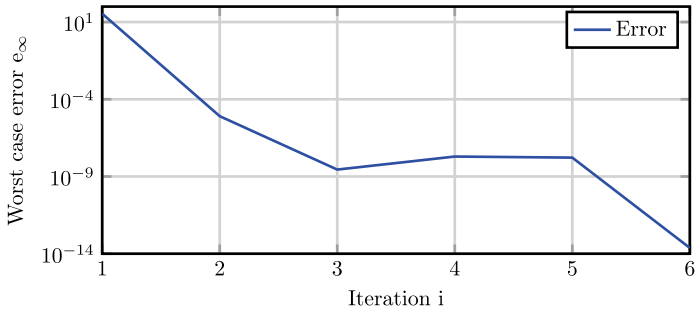

Figure 8.3: Example of Section 8.3.3: worst-case fitting error e_∞ as a function of iteration counter i. The last iteration ($i = 6$) was performed with fixed poles.

obtained:

$$Z(\jmath\omega) = \frac{\mathscr{F}\{p(t)\}}{\mathscr{F}\{q(t)\}}, \tag{8.32}$$

where $\mathscr{F}\{.\}$ denotes the Fourier transform. Impedance was computed at $\bar{k} = 11$ frequency points $\omega_k = 2\pi(k-1)f_0$ for $k = 1,\ldots,11$, where $f_0 = 2.54\,\text{Hz} = 152.4\,\text{beats/min}$ corresponds to the heart rate of the patient. The authors of [71] estimate that the impedance measurements are affected by uncertainty with a relative standard deviation that ranges between 0.66 % to 14.5 % depending on frequency. The relative standard deviation was normalized to $|Z(0)|$.

We apply VF to the impedance samples to obtain a closed-form model relating aortic pressure and flow rate. The limited number of available samples and their uncertainty make the identification of an accurate model challenging. We use this nontrivial scenario to explore the relation between number and quality of the available samples, model order $\bar{n}$, and accuracy. Vector Fitting was applied to the given samples four times with model order $\bar{n}$ increasing from 2 to 8 in steps of 2. Figure 8.4 compares the magnitude and phase of the identified model to the original impedance samples. We can see that the $\bar{n} = 2$ model captures the overall trend of the impedance. However, it fails to resolve the increase in impedance at $f = 12.7\,\text{Hz}$ and the associated phase variation. Increasing order to 4 or 6 resolves that feature and provides higher accuracy. Further increasing order $\bar{n}$ to 8 leads to a model which matches closely most given samples, but has a sharp and high peak at $f = 12.3\,\text{Hz}$. This unrealistic behavior in-between the given samples is typical of an overfitting scenario, where the sought model has too many degrees of freedom, which can be hardly estimated from the information contained in the available samples. Although still solvable, the conditioning number of (8.21) degrades. The system solution, which gives the model coefficients, becomes very sensitive to the noise superimposed to the given samples. The source codes related to this example can be downloaded from [75].

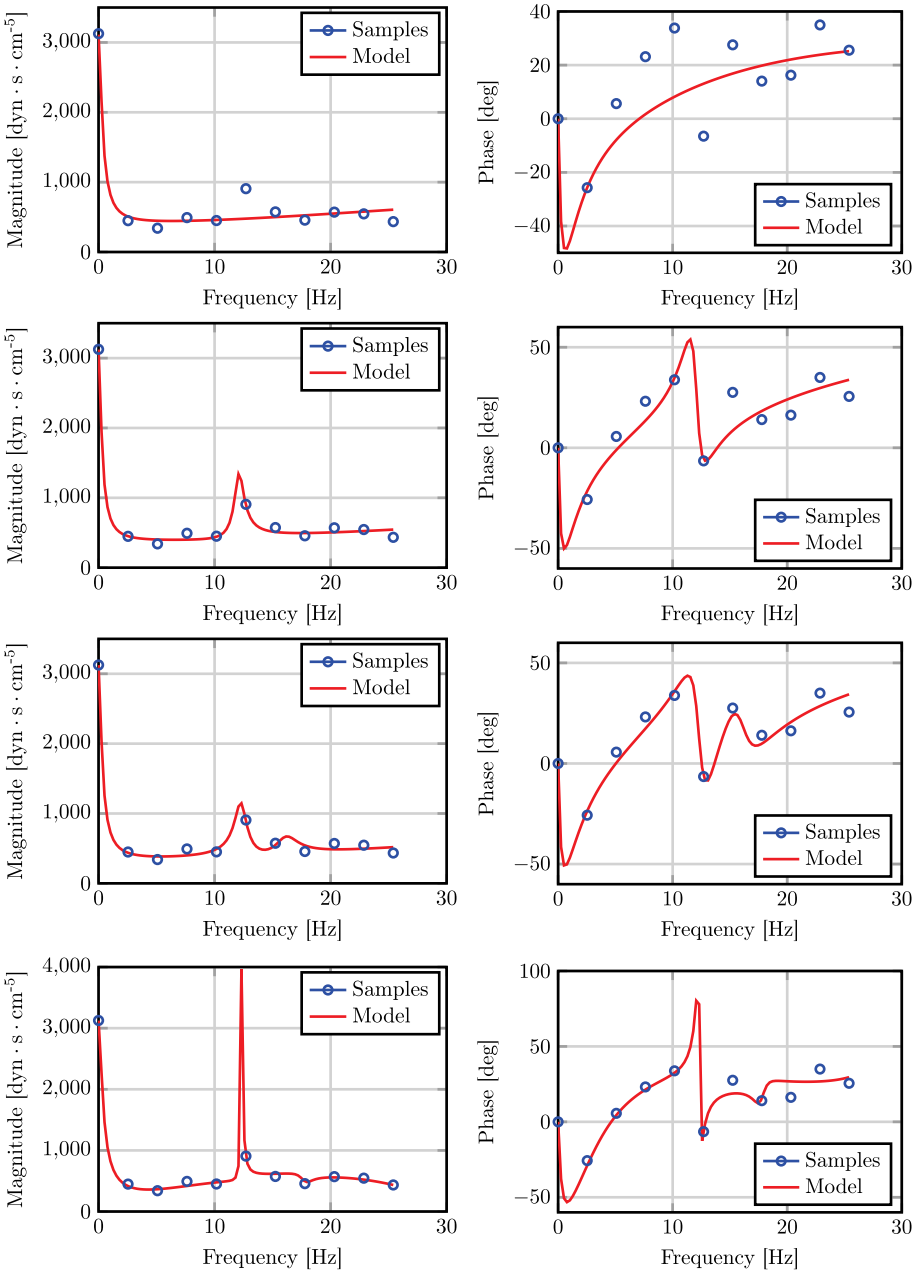

Figure 8.4: Impedance seen into the ascending aorta of the pediatric patient considered in Section 8.3.4: measured samples (circles) and response of four different VF models (dashed lines) of order $\bar{n} = 2, 4, 6, 8$ (from top to bottom).

8.3.5 The multi-input multi-output case

The VF algorithm presented in Section 8.3.2 for the single-input single-output case can be easily extended to the general case of a system with $\bar{m}$ inputs and $\bar{q}$ outputs. In this case, the given samples are $\bar{q} \times \bar{m}$ complex matrices H_k, and we denote their (q, m) entry as $H_{k,qm}$. The model transfer function is now defined as

$$\widetilde{H}^{(i)}(s) = R_0^{(i)} + \sum_{n=1}^{\bar{n}} \frac{R_n^{(i)}}{s - p_n^{(i-1)}}, \tag{8.33}$$

where $R_n^{(i)} \in \mathbb{C}^{\bar{q} \times \bar{m}}$. In (8.33), the same poles $p_n^{(i-1)}$ are used for all elements of matrix $\widetilde{H}^{(i)}(s)$. This choice is appropriate when modeling linear dynamical systems, since it is known that the poles of each transfer function entry are a subset of a common set of poles shared by all transfer function elements. The physical justification of this fact is that poles are related to the natural modes of the system, which are a property of the system itself and not of individual entries of its transfer function. In other communities, natural modes are referred to as resonances or eigenmodes of the system. When VF is applied to model transfer functions not related to the same physical system, one should use distinct poles for different elements of (8.33). This scenario is discussed in [35], which also elaborates on the computational implications of this choice.

In the multi-input multi-output case, weighting function $w^{(i)}(s)$ remains defined by (8.15). Since the transfer function (8.33) is now matrix-valued, VF aims to minimize the error functional

$$e^2 = \frac{1}{\bar{k}\bar{q}\bar{m}} \sum_{k=1}^{\bar{k}} \left\| H_k - \widetilde{H}(\jmath\omega_k) \right\|_F^2, \tag{8.34}$$

where $\|.\|_F$ denotes the Frobenius norm, which for $A \in \mathbb{C}^{\bar{q} \times \bar{m}}$ is defined as

$$\|A\|_F = \sqrt{\sum_{q=1}^{\bar{q}} \sum_{m=1}^{\bar{m}} |A_{qm}|^2}. \tag{8.35}$$

From (8.35), we see that the square of the Frobenius norm is simply equal to the sum of the squared magnitudes of each entry. Therefore, minimizing (8.34) means minimizing the sum of the squared error between each sample $H_{k,qm}$ and the corresponding entry of (8.33).

The minimization of (8.34) is a nonlinear least-squares problem, which VF solves iteratively by working on the linearized error [42]

$$\left(e_{SK}^{(i)}\right)^2 = \frac{1}{\bar{k}\bar{q}\bar{m}} \sum_{k=1}^{\bar{k}} \left\| H_k \left(1 + \sum_{n=1}^{\bar{n}} \frac{w_n^{(i)}}{\jmath\omega_k - p_n^{(i-1)}} \right) - \left(R_0^{(i)} + \sum_{n=1}^{\bar{n}} \frac{R_n^{(i)}}{\jmath\omega_k - p_n^{(i-1)}} \right) \right\|_F^2. \tag{8.36}$$

As in the single-input single-output case, we can see that (8.36) uses weighting function $w^{(i)}(s)$ to offset the error introduced by using the previous poles estimate in the

denominators. Minimizing (8.36) is equivalent to solving, in the least-squares sense, the system of equations

$$R_{0,qm}^{(i)} + \sum_{n=1}^{\bar{n}} \frac{R_{n,qm}^{(i)}}{\jmath\omega_k - p_n^{(i-1)}} - H_{k,qm} \sum_{n=1}^{\bar{n}} \frac{w_n^{(i)}}{\jmath\omega_k - p_n^{(i-1)}} = H_{k,qm} \tag{8.37}$$

for $k = 1, \dots, \bar{k}$, $q = 1, \dots, \bar{q}$ and $m = 1, \dots, \bar{m}$. In matrix form, equations (8.37) read

$$\begin{bmatrix} \Phi_0^{(i)} & 0 & \dots & 0 & -D_{H_{11}}\Phi_1^{(i)} \\ 0 & \Phi_0^{(i)} & \ddots & \vdots & -D_{H_{21}}\Phi_1^{(i)} \\ \vdots & \ddots & \ddots & 0 & \vdots \\ 0 & \dots & 0 & \Phi_0^{(i)} & -D_{H_{\bar{q}\bar{m}}}\Phi_1^{(i)} \end{bmatrix} \begin{bmatrix} c_{H_{11}}^{(i)} \\ c_{H_{21}}^{(i)} \\ \vdots \\ c_{H_{\bar{q}\bar{m}}}^{(i)} \\ c_w^{(i)} \end{bmatrix} = \begin{bmatrix} V_{H_{11}} \\ V_{H_{21}} \\ \vdots \\ V_{H_{\bar{q}\bar{m}}} \end{bmatrix}, \tag{8.38}$$

where $D_{H_{qm}}$ and $V_{H_{qm}}$ are, respectively, a diagonal matrix and a column vector formed by all samples $H_{k,qm}$ for $k = 1, \dots, \bar{k}$. In the unknown vector of (8.38),

$$c_{H_{qm}}^{(i)} = \begin{bmatrix} R_{0,qm}^{(i)} & \dots & R_{\bar{n},qm}^{(i)} \end{bmatrix}^T, \tag{8.39}$$

and $c_w^{(i)}$ is defined by (8.26). System (8.38) is solved in step 4 of Algorithm 8.1. In step 8, a tentative final fitting of the model is performed, assuming fixed poles and determining only a new estimate of residues $R_n^{(i+1)}$. This step can be achieved by solving

$$\Phi_0^{(i+1)} c_{H_{qm}}^{(i+1)} = V_{H_{qm}}, \tag{8.40}$$

for $q = 1, \dots, \bar{q}$ and $m = 1, \dots, \bar{m}$.

8.3.6 The fast Vector Fitting algorithm

As the number of inputs $\bar{m}$ and outputs $\bar{q}$ increases, the computational cost of solving (8.38) can quickly become unsustainable. As technology evolves, this scenario arises more frequently, as engineers need to model systems of increasing complexity, either in terms of dynamic order or number of inputs and outputs. For example, a modern server processor has about 2,000 pins, which are connected to the motherboard by a dense network of tiny wires realized on the chip package. Seen as an input–output system, this network will have about 4,000 inputs and outputs, half where the network connects to the motherboard, and half where the network is connected to the silicon die. The need to predict electromagnetic interference in this dense and intricate network of wires calls for scalable algorithms to create reduced order models for systems where the number of inputs $\bar{m}$ and outputs $\bar{q}$ can be several thousands [70, 8].

The Fast VF algorithm [22, 47] significantly reduces the cost of solving (8.38) for multi-input and multi-output systems. Savings are achieved by exploiting the block

structure of (8.38) and the fact that, of the solution vector of (8.38), only $c_w^{(i)}$ is actually needed to compute the new poles estimate (8.18). A least-squares problem in the form of (8.38) can be efficiently solved by first performing the QR decompositions [29, 9, 86, 22]

$$\begin{bmatrix} \Phi_0^{(i)} & -D_{H_{qm}} \Phi_1^{(i)} \end{bmatrix} = \begin{bmatrix} \mathcal{Q}_{qm}^1 & \mathcal{Q}_{qm}^2 \end{bmatrix} \begin{bmatrix} \mathcal{R}_{qm}^{11} & \mathcal{R}_{qm}^{12} \\ 0 & \mathcal{R}_{qm}^{22} \end{bmatrix}, \tag{8.41}$$

for $q = 1, \ldots, \bar{q}$ and $m = 1, \ldots, \bar{m}$. Then a reduced system is formed [22]:

$$\begin{bmatrix} \mathcal{R}_{11}^{22} \\ \mathcal{R}_{21}^{22} \\ \vdots \\ \mathcal{R}_{\bar{q}\bar{m}}^{22} \end{bmatrix} c_w^{(i)} = \begin{bmatrix} (\mathcal{Q}_{11}^2)^H V_{H_{11}} \\ (\mathcal{Q}_{21}^2)^H V_{H_{21}} \\ \vdots \\ (\mathcal{Q}_{\bar{q}\bar{m}}^2)^H V_{H_{\bar{q}\bar{m}}} \end{bmatrix}, \tag{8.42}$$

where ^{H} denotes the conjugate transpose, also known as Hermitian transpose. System (8.42) is solved in the least-squares sense to determine $c_w^{(i)}$, and compute the new poles estimate with (8.18). Computational savings arise from the fact that the size of the matrices involved in (8.41) and (8.42) is much lower than the size of the coefficient matrix in (8.38). Furthermore, since the $\bar{q}\bar{m}$ QR decompositions (8.41) are independent, they can be performed in parallel [14]. The Fast VF algorithm with parallelization can identify reduced models for systems with hundreds of inputs and outputs in minutes [35]. A pseudo-code of a real-valued implementation of the Fast VF algorithm will be given in Section 8.3.8.

Several other ideas were proposed to increase VF scalability for large input and output counts. In VF with compression, samples H_k are "compressed" with a singular value decomposition reducing the cost of the subsequent fitting [37] and passivity enforcement steps [63]. The Loewner method [52, 46], which is an alternative to VF for the data-driven modeling of linear systems, was also shown to scale favorably with respect to the number of inputs and outputs. This class of techniques is the subject of Chapter 6 of this volume.

8.3.7 Example: modeling of a multiport interconnect on a printed circuit board

Vector Fitting is extensively used by electronic designers to model how high-speed digital signals propagate over a printed circuit board, and design the system accordingly. We consider the structure shown in Figure 8.5, which consists of several copper traces realized on the top face of a high-performance printed circuit board (Wild River Technology CMP-28 [88]). This structure mimics, in a simplified way, the multiwire buses that may connect the CPU and memory of a high-performance server. At the end of

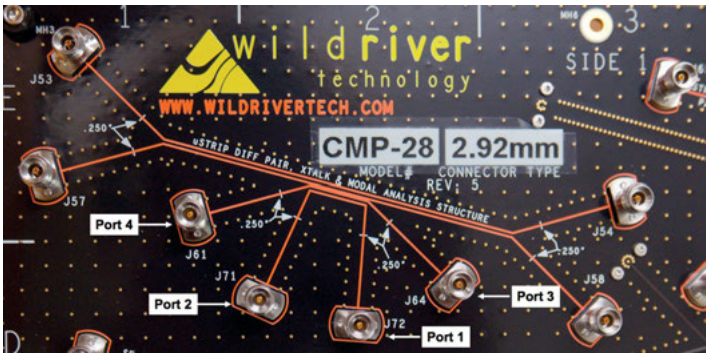

Figure 8.5: Interconnect network on a printed circuit board considered in Section 8.3.7. The four measurement ports of the vector network analyzer were connected as shown in the Figure.

each trace, an electrical port is defined between the trace endpoint and a reference point on the ground plane underneath. The port is defined where the CPU or memory chip would be connected. In the test system, a high-frequency connector was installed at each port allowing the user to inject a signal from each port, and observe the signal received at the other ports.

In this example, we consider the two lower traces in Figure 8.5, which have connectors J72, J71, J64, J61 soldered at their ends. The scattering matrix $H(\jmath\omega)$ of this 4-port device was measured from 10 MHz to 40 GHz in steps of 10 MHz with a Keysight N5227A vector network analyzer (courtesy of Fadime Bekmambetova, University of Toronto). In the scattering representation, input $U_m(\jmath\omega)$ is the amplitude of the electromagnetic wave injected into port m by the instrument. Output $Y_q(\jmath\omega)$ is the amplitude of the wave received at port q. The scattering representation is commonly used at high frequency since it can be measured more accurately compared to the impedance or admittance representations used at low frequency.

A commercial implementation of the VF algorithm (IdEM, Dassault Systemes) was used to generate a reduced order model from the measured samples (courtesy of Prof. Stefano Grivet-Talocia, Politecnico di Torino). Figure 8.6 compares the VF model response to the original samples for the $(1,2)$ element of the scattering matrix. This response is the ratio between the amplitude of the wave received at one end of the trace (port 1) and the amplitude of the wave injected at the other end (port 2). We see that, as frequency increases, the received signal is progressively weaker, due to higher attenuation. The agreement between the VF model and the samples is excellent over the entire frequency range spanned by the measured data. Figure 8.7 compares the model response to the measured samples for the $(1,3)$ entry of the scattering matrix, which describes the signal received on the lower copper trace in Figure 8.5 when only the upper trace is excited. This coefficient is about 25 times smaller than the $(1,2)$ coefficient, since the two traces are not directly connected, and any coupling is due to

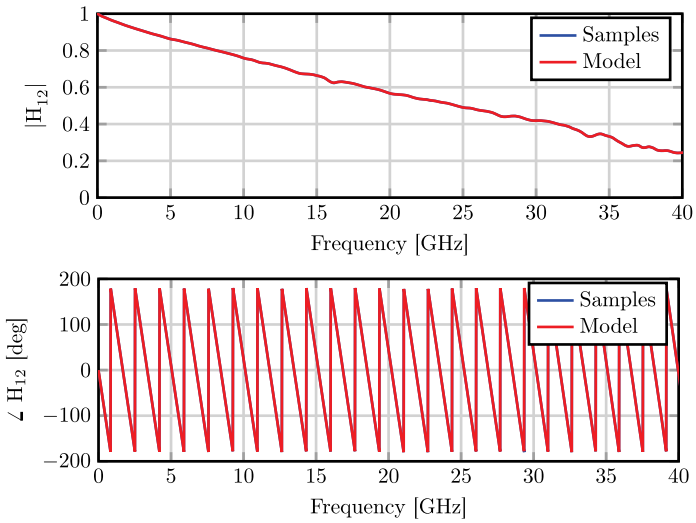

Figure 8.6: Example of Section 8.3.7: comparison between samples $H_{k,12}$ and corresponding VF model response.

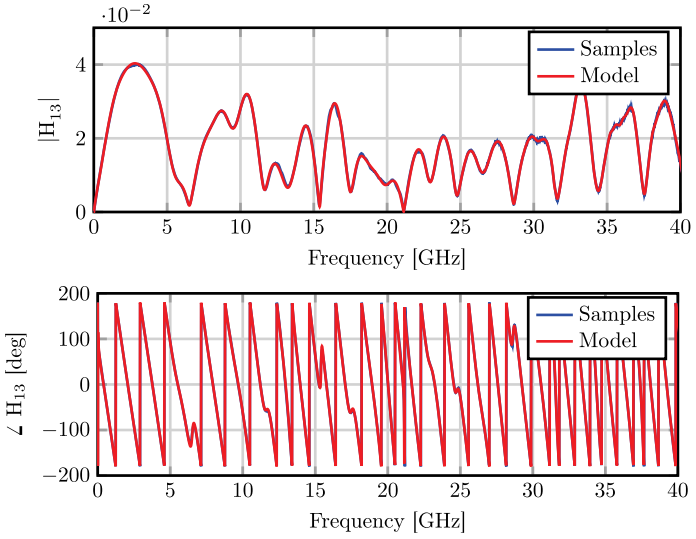

Figure 8.7: Example of Section 8.3.7: comparison between samples $H_{k,13}$ and corresponding VF model response.

electromagnetic interference. We can see that the VF model approximates this small entry very accurately.

Figure 8.8 plots the samples-model error $e_{SK}^{(i)}$ as a function of i, together with the order $\bar{n}$ used by VF at each iteration. In this example, the order is adapted throughout

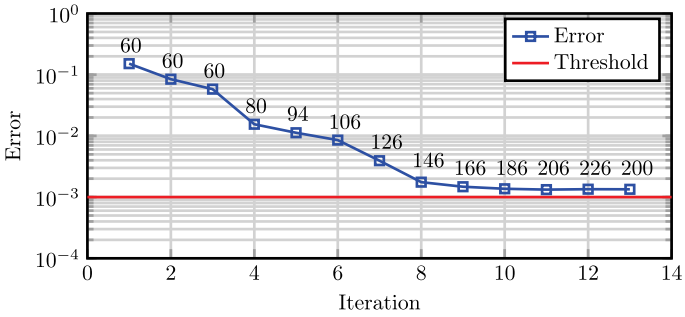

Figure 8.8: Example of Section 8.3.7: VF error as a function of iteration, compared to the desired error level. Labels indicate the order $\bar{n}$ used by VF at each iteration.

iterations with the adding and skimming process [33] described in Section 8.3.11.1. We observe that VF is able to progressively reduce the error throughout iterations, but convergence is slower than in the analytical example of Section 8.3.3. This happens because of two reasons. First, this implementation of VF adaptively determines order $\bar{n}$ in a single run, without requiring the user to determine a suitable $\bar{n}$ with multiple VF runs. Second, some noise is unavoidably present in the experimental measurements, which slows down convergence, and prevents VF from reducing the fitting error below 10^{-3}. Indeed, we can see that VF is unable to increase model accuracy after the 10th iteration. Ultimately, VF delivers a reduced model with an error of $1.34 \cdot 10^{-3}$, which is adequate for most design purposes.

8.3.8 A real-valued formulation of VF and fast VF

In most systems of practical interest, input $u(t)$ and output $y(t)$ are real-valued. Consequently, poles p_n and residues R_n are expected to be either real or in complex conjugate pairs. Because of round-off errors, the VF algorithm described so far may not ensure this realness condition. In this section, we describe a real-valued version of Fast VF which can be implemented in real arithmetics, and will ensure the realness condition by construction. The pseudo-code of the described algorithm is given in Algorithm 8.2. An open-source implementation of this algorithm, which closely follows the notation and pseudo-code in this chapter, can be downloaded from [75].

To ensure complex conjugate poles and residues, we redefine model (8.33) as

$$\widetilde{H}^{(i)}(s) = R_0^{(i)} + \sum_{n=1}^{\bar{n}_r} \frac{R_n^{(i)}}{s - p_n^{(i-1)}} + \sum_{n=\bar{n}_r+1}^{\bar{n}_r+\bar{n}_c} \left[\frac{R_n^{(i)}}{s - p_n^{(i-1)}} + \frac{(R_n^{(i)})^*}{s - (p_n^{(i-1)})^*} \right], \tag{8.43}$$

where $\bar{n}_r$ is the number of real poles and $\bar{n}_c$ is the number of pairs of complex conjugate poles, for a total order $\bar{n} = \bar{n}_r + 2\bar{n}_c$. In (8.43), we force $R_n^{(i)} \in \mathbb{R}^{\bar{q}\times\bar{m}}$ for $n = 0, \ldots, \bar{n}_r$. The

VF weighting function (8.15) is redefined in a similar fashion as

$$w^{(i)}(s) = 1 + \sum_{n=1}^{\bar{n}_r} \frac{w_n^{(i)}}{s - p_n^{(i-1)}} + \sum_{n=\bar{n}_r+1}^{\bar{n}_r+\bar{n}_c} \left[\frac{w_n^{(i)}}{s - p_n^{(i-1)}} + \frac{(w_n^{(i)})^*}{s - (p_n^{(i-1)})^*} \right], \tag{8.44}$$

where $w_n^{(i)} \in \mathbb{R}$ for $n = 1, \ldots, \bar{n}_r$. Using (8.43) and (8.44), and following the steps in Section 8.3.5, one can arrive at a least-squares system in the same form as (8.38)

$$\begin{bmatrix} \Phi_0^{(i)} & 0 & \dots & 0 & -D_{H_{11}}\Phi_1^{(i)} \\ 0 & \Phi_0^{(i)} & \ddots & \vdots & -D_{H_{21}}\Phi_1^{(i)} \\ \vdots & \ddots & \ddots & 0 & \vdots \\ 0 & \dots & 0 & \Phi_0^{(i)} & -D_{H_{\bar{q}\bar{m}}}\Phi_1^{(i)} \end{bmatrix} \begin{bmatrix} c_{H_{11}}^{(i)} \\ c_{H_{21}}^{(i)} \\ \vdots \\ c_{H_{qm}}^{(i)} \\ c_w^{(i)} \end{bmatrix} = \begin{bmatrix} V_{H_{11}} \\ V_{H_{21}} \\ \vdots \\ V_{H_{\bar{q}\bar{m}}} \end{bmatrix}, \tag{8.45}$$

but where we take as unknowns the real and imaginary part of each residue in (8.43)

$$c_{H_{qm}}^{(i)} = \begin{bmatrix} R_{0,qm}^{(i)} & \dots & R_{\bar{n}_r,qm}^{(i)} & \mathrm{Re}\{R_{\bar{n}_r+1,qm}^{(i)}\} & \mathrm{Im}\{R_{\bar{n}_r+1,qm}^{(i)}\} & \dots \end{bmatrix}^T, \tag{8.46}$$

and the real and imaginary part of each residue of the weighting function (8.44)

$$c_w^{(i)} = \begin{bmatrix} w_1^{(i)} & \dots & w_{\bar{n}_r}^{(i)} & \mathrm{Re}\{w_{\bar{n}_r+1}^{(i)}\} & \mathrm{Im}\{w_{\bar{n}_r+1}^{(i)}\} & \dots \end{bmatrix}^T. \tag{8.47}$$

This choice of unknowns will ensure that complex residues always come in conjugate pairs. The coefficient matrices $\Phi_0^{(i)}$ and $\Phi_1^{(i)}$ in (8.45) are given by

$$\Phi_0^{(i)} = \begin{bmatrix} 1_{\bar{k}} & \Phi_r^{(i)} & \Phi_c^{(i)} \end{bmatrix}, \tag{8.48}$$

$$\Phi_1^{(i)} = \begin{bmatrix} \Phi_r^{(i)} & \Phi_c^{(i)} \end{bmatrix}, \tag{8.49}$$

where $1_{\bar{k}}$ is a $\bar{k} \times 1$ vector of ones, and

$$\Phi_r^{(i)} = \begin{bmatrix} \frac{1}{\jmath\omega_1 - p_1^{(i-1)}} & \dots & \frac{1}{\jmath\omega_1 - p_{\bar{n}_r}^{(i-1)}} \\ \vdots & \ddots & \vdots \\ \frac{1}{\jmath\omega_{\bar{k}} - p_1^{(i-1)}} & \dots & \frac{1}{\jmath\omega_{\bar{k}} - p_{\bar{n}_r}^{(i-1)}} \end{bmatrix}, \tag{8.50}$$

$$\Phi_c^{(i)} = \begin{bmatrix} \frac{1}{\jmath\omega_1 - p_{\bar{n}_r+1}^{(i-1)}} + \frac{1}{\jmath\omega_1 - (p_{\bar{n}_r+1}^{(i-1)})^*} & \frac{\jmath}{\jmath\omega_1 - p_{\bar{n}_r+1}^{(i-1)}} - \frac{\jmath}{\jmath\omega_1 - (p_{\bar{n}_r+1}^{(i-1)})^*} & \dots \\ \vdots & \vdots & \\ \frac{1}{\jmath\omega_{\bar{k}} - p_{\bar{n}_r+1}^{(i-1)}} + \frac{1}{\jmath\omega_{\bar{k}} - (p_{\bar{n}_r+1}^{(i-1)})^*} & \frac{\jmath}{\jmath\omega_{\bar{k}} - p_{\bar{n}_r+1}^{(i-1)}} - \frac{\jmath}{\jmath\omega_{\bar{k}} - (p_{\bar{n}_r+1}^{(i-1)})^*} & \dots \end{bmatrix}. \tag{8.51}$$

Although (8.45) has real unknowns, its coefficients matrix and right hand side are still complex-valued. To remedy this issue, we write the real and imaginary part of each

equation separately

$$\begin{bmatrix} \mathrm{Re}\{\Phi_0^{(i)}\} & 0 & \dots & 0 & -\mathrm{Re}\{D_{H_{11}}\Phi_1^{(i)}\} \\ \mathrm{Im}\{\Phi_0^{(i)}\} & 0 & \dots & 0 & -\mathrm{Im}\{D_{H_{11}}\Phi_1^{(i)}\} \\ \vdots & & & \vdots & \vdots \\ 0 & \dots & 0 & \mathrm{Re}\{\Phi_0^{(i)}\} & -\mathrm{Re}\{D_{H_{\bar{q}\bar{m}}}\Phi_1^{(i)}\} \\ 0 & \dots & 0 & \mathrm{Im}\{\Phi_0^{(i)}\} & -\mathrm{Im}\{D_{H_{\bar{q}\bar{m}}}\Phi_1^{(i)}\} \end{bmatrix} \begin{bmatrix} c_{H_{11}}^{(i)} \\ \vdots \\ c_{H_{\bar{q}\bar{m}}}^{(i)} \\ c_w^{(i)} \end{bmatrix} = \begin{bmatrix} \mathrm{Re}\{V_{H_{11}}\} \\ \mathrm{Im}\{V_{H_{11}}\} \\ \vdots \\ \mathrm{Re}\{V_{H_{\bar{q}\bar{m}}}\} \\ \mathrm{Im}\{V_{H_{\bar{q}\bar{m}}}\} \end{bmatrix}. \tag{8.52}$$

The obtained system, which has real coefficients and unknowns will ensure, by construction, that model poles and residues are either real or complex conjugate. Due to its block structure, system (8.52) can be efficiently solved with the Fast VF approach discussed in Section 8.3.6. In step 5 of Algorithm 8.2, the QR decompositions

$$\begin{bmatrix} \mathrm{Re}\{\Phi_0^{(i)}\} & -\mathrm{Re}\{D_{H_{qm}}\Phi_1^{(i)}\} \\ \mathrm{Im}\{\Phi_0^{(i)}\} & -\mathrm{Im}\{D_{H_{qm}}\Phi_1^{(i)}\} \end{bmatrix} = \begin{bmatrix} \mathcal{Q}_{qm}^{11} & \mathcal{Q}_{qm}^{12} \\ \mathcal{Q}_{qm}^{21} & \mathcal{Q}_{qm}^{22} \end{bmatrix} \begin{bmatrix} \mathcal{R}_{qm}^{11} & \mathcal{R}_{qm}^{12} \\ 0 & \mathcal{R}_{qm}^{22} \end{bmatrix}, \tag{8.53}$$

are computed for $q = 1, \dots, \bar{q}$ and $m = 1, \dots, \bar{m}$. Then, in step 5, the reduced system

$$\begin{bmatrix} \mathcal{R}_{11}^{22} \\ \mathcal{R}_{21}^{22} \\ \vdots \\ \mathcal{R}_{\bar{q}\bar{m}}^{22} \end{bmatrix} c_w^{(i)} = \begin{bmatrix} (\mathcal{Q}_{11}^{12})^T \mathrm{Re}\{V_{H_{11}}\} + (\mathcal{Q}_{11}^{22})^T \mathrm{Im}\{V_{H_{11}}\} \\ (\mathcal{Q}_{21}^{12})^T \mathrm{Re}\{V_{H_{21}}\} + (\mathcal{Q}_{21}^{22})^T \mathrm{Im}\{V_{H_{21}}\} \\ \vdots \\ (\mathcal{Q}_{\bar{q}\bar{m}}^{12})^T \mathrm{Re}\{V_{H_{\bar{q}\bar{m}}}\} + (\mathcal{Q}_{\bar{q}\bar{m}}^{22})^T \mathrm{Im}\{V_{H_{\bar{q}\bar{m}}}\} \end{bmatrix} \tag{8.54}$$

is solved in the least-squares sense to determine $c_w^{(i)}$ and compute the new poles estimate with the real-valued counterpart of (8.18), which reads [35]

$$\{p_n^{(i)}\} = \mathrm{eig}(A^{(i-1)} - b_w (c_w^{(i)})^T), \tag{8.55}$$

with $A^{(i-1)} = \mathrm{diag}\{p_1^{(i-1)}, \dots, p_{\bar{n}_r}^{(i-1)}, \Pi_{\bar{n}_r+1}^{(i-1)}, \dots, \Pi_{\bar{n}_r+\bar{n}_c}^{(i-1)}\}$ being a block diagonal matrix formed by the real poles and, for complex conjugate pairs, by the blocks

$$\Pi_n^{(i-1)} = \begin{bmatrix} \mathrm{Re}\{p_n^{(i-1)}\} & \mathrm{Im}\{p_n^{(i-1)}\} \\ -\mathrm{Im}\{p_n^{(i-1)}\} & \mathrm{Re}\{p_n^{(i-1)}\} \end{bmatrix}. \tag{8.56}$$

In (8.55), b_w is a $\bar{n} \times 1$ vector with the first $\bar{n}_r$ entries set to one, followed by a $[2, 0]^T$ block for each pair of complex conjugate poles.

Once poles have been estimated, a first convergence test is performed in step 8 using (8.27). If the test is passed, in step 9 of Algorithm 8.2 we fit the residues of the final model, solving in the least-squares sense

$$\begin{bmatrix} \mathrm{Re}\{\Phi_0^{(i+1)}\} \\ \mathrm{Im}\{\Phi_0^{(i+1)}\} \end{bmatrix} c_{H_{qm}}^{(i+1)} = \begin{bmatrix} \mathrm{Re}\{V_{H_{qm}}\} \\ \mathrm{Im}\{V_{H_{qm}}\} \end{bmatrix}, \tag{8.57}$$

for $q = 1, \dots, \bar{q}$ and $m = 1, \dots, \bar{m}$. The second and final convergence test is performed in step 11 of Algorithm 8.2.

Algorithm 8.2: Fast Vector Fitting, real-valued implementation.

Require: response samples H_k, corresponding frequencies ω_k $(k = 1, \ldots, \bar{k})$
Require: desired model order $\bar{n}$
Require: maximum number of iterations $i_{\max}$
set initial poles $p_n^{(0)}$ according to (8.19) or (8.20).
$i \leftarrow 1$
while $i \leq i_{\max}$ **do**
    Compute QR decompositions (8.53)
    Solve (8.54) in the least-squares sense
    Compute the new poles estimate $p_n^{(i)}$ with (8.55)
    Enforce poles stability with (8.71), if desired ▷ Stability enforcement
    if (8.27) is true **then** ▷ First convergence test
        Solve (8.57) in the least-squares sense ▷ Tentative final fitting
        Compute fitting error e with (8.34)
        if $e \leq \varepsilon_H$ **then** ▷ Second convergence test
            $\widetilde{H}(s) = \widetilde{H}^{(i+1)}(s)$
            return Success!
        end if
    end if
    $i \leftarrow i + 1$
end while
return Failure: maximum number of iterations reached.

8.3.9 Model realization

The real-valued formulation of VF, discussed in Section 8.3.8, produces a reduced model in the form

$$\widetilde{H}(s) = R_0 + \sum_{n=1}^{\bar{n}_r} \frac{R_n}{s - p_n} + \sum_{n=\bar{n}_r+1}^{\bar{n}_r+\bar{n}_c} \left[\frac{R_n}{s - p_n} + \frac{R_n^*}{s - p_n^*} \right], \tag{8.58}$$

which can be easily converted into a variety of equivalent representations to facilitate its use in different simulation scenarios. Expression (8.58) is known as pole-residue form of the transfer function. This form is the most convenient when the model will be used in frequency domain analyses, since it minimizes the computational cost of evaluating $H(\jmath\omega)$.

For time domain analyses, such as transient simulations, expression (8.58) can be converted into the time domain with the inverse Laplace transform, which yields

$$\widetilde{h}(t) = R_0 + \sum_{n=1}^{\bar{n}_r} R_n e^{p_n t} + \sum_{n=\bar{n}_r+1}^{\bar{n}_r+\bar{n}_c} \left[2R_n' e^{p_n' t} \cos(p_n'' t) - 2R_n'' e^{p_n' t} \sin(p_n'' t) \right] \tag{8.59}$$

for $t \geq 0$, where $p_n' = \text{Re}\{p_n\}$, $p_n'' = \text{Im}\{p_n\}$, $R_n' = \text{Re}\{R_n\}$ and $R_n'' = \text{Im}\{R_n\}$. In (8.59), $\widetilde{h}(t)$ denotes the impulse response of the model. This form is particularly convenient in transient simulators based on convolutions like (8.1). While computing convolution integrals is in general very expensive, when an impulse response has the form (8.59), convolution can be computed very quickly using recursive formulas [35].

While convolutional simulators are prominent in selected applications, the majority of transient simulators is based on the solution of differential equations, and cannot handle (8.59) directly. To overcome this issue, we can represent (8.58) through a set of differential equations in state-space form

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t), \\ y(t) = Cx(t) + Du(t). \end{cases} \tag{8.60}$$

System (8.60) is constructed in such a way that the transfer function between input $u(t)$ and output $y(t)$ is (8.58). Given a transfer function, there are infinitely-many systems (8.60) that meet this criterion, known as *realizations* of $H(s)$. We present a popular realization, due to Gilbert [27], and refer the reader to [35] for a comprehensive description of how VF models can be realized.

For reasons that will become clear later on, the Gilbert realization process begins with the truncated singular value decomposition [28] of residues R_n

$$R_n = U_n \Sigma_n V_n^H \quad \text{for } n = 1, \ldots, \bar{n}_r + \bar{n}_c, \tag{8.61}$$

where $\Sigma_n = \text{diag}\{\sigma_{n,1}, \ldots, \sigma_{n,\rho_n}\}$ is a diagonal matrix collecting all nonzero singular values of R_n, and ρ_n is the rank of R_n. Matrices $U_n \in \mathbb{C}^{\bar{q} \times \rho_n}$ and $V_n \in \mathbb{C}^{\bar{m} \times \rho_n}$ are formed by the left and right singular vectors of R_n, respectively. Given (8.61), we can express the partial fractions in (8.58) associated to real poles as

$$\frac{R_n}{s - p_n} = U_n \Sigma_n \frac{I_{\rho_n}}{s - p_n} V_n^T = C_n (s I_{\rho_n} - A_n)^{-1} B_n, \tag{8.62}$$

for $n = 1, \ldots, \bar{n}_r$. In (8.62), I_{ρ_n} is the identity matrix of size $\rho_n \times \rho_n$, $C_n = U_n \Sigma_n$, $A_n = p_n I_{\rho_n}$, and $B_n = V_n^T$. For complex poles, we can derive an equivalent expression for the sum of the two conjugate partial fractions [35]

$$\frac{R_n}{s - p_n} + \frac{R_n^*}{s - p_n^*} = C_n (s I_{2\rho_n} - A_n)^{-1} B_n, \tag{8.63}$$

for $n = \bar{n}_r + 1, \ldots, \bar{n}_r + \bar{n}_c$, where

$$A_n = \begin{bmatrix} p_n' I_{\rho_n} & p_n'' I_{\rho_n} \\ -p_n'' I_{\rho_n} & p_n' I_{\rho_n} \end{bmatrix} \quad B_n = 2 \begin{bmatrix} \text{Re}\{V_n^H\} \\ \text{Im}\{V_n^H\} \end{bmatrix} \tag{8.64}$$

$$C_n = \begin{bmatrix} \text{Re}\{U_n \Sigma_n\} & \text{Im}\{U_n \Sigma_n\} \end{bmatrix}. \tag{8.65}$$

Expressions (8.62) and (8.63) allow us to rewrite (8.58) as

$$\widetilde{H}(s) = D + C(sI_N - A)^{-1}B, \tag{8.66}$$

where

$$A = \begin{bmatrix} A_1 & & \\ & \ddots & \\ & & A_{\bar{n}_r+\bar{n}_c} \end{bmatrix}, \quad B = \begin{bmatrix} B_1 \\ \vdots \\ B_{\bar{n}_r+\bar{n}_c} \end{bmatrix}, \tag{8.67}$$

$$C = \begin{bmatrix} C_1 & \dots & C_{\bar{n}_r+\bar{n}_c} \end{bmatrix}, \quad D = R_0. \tag{8.68}$$

Since (8.66) is the transfer function of (8.60), equations (8.67) and (8.68) provide the coefficient matrices of a state space realization (8.60) of the transfer function (8.58) produced by VF. The order of (8.66) is

$$N = \sum_{n=1}^{\bar{n}_r} \rho_n + 2 \sum_{n=\bar{n}_r+1}^{\bar{n}_r+\bar{n}_c} \rho_n \tag{8.69}$$

and can be shown to be minimal [27]. This property stems from the singular value decompositions (8.61), which reveal the rank ρ_n of each residue R_n. If those singular value decompositions are not performed, a realization of order $\bar{n}\bar{m}$ is obtained. This realization may not be minimal, and may contain states that are not controllable, not observable, or both, as discussed in Chapter 2 of this volume.

In addition to the forms presented in this section, the VF model (8.58) can be converted to a variety of additional forms, including equivalent electric circuits [5, 35] for seamless integration into any circuit simulator.

8.3.10 Stability, causality and passivity enforcement

Most systems of practical interest are stable, and the real part of their poles is either negative or zero. One would expect that, given noise-free samples of their frequency response, VF will produce a model with stable poles satisfying

$$\mathrm{Re}\{p_n\} \leq 0 \quad \forall n. \tag{8.70}$$

Unfortunately, this is not guaranteed, since round-off errors may indeed push a few poles into the right half of the complex plane, making the VF model unstable.

Condition (8.70) is essentially mandatory for time domain simulations, since otherwise results will diverge. The standard practice is to enforce stability during VF iterations. After computing the new poles estimate $p_n^{(i)}$ with (8.18), the following rule is applied:

$$p_n^{(i)} = \begin{cases} p_n^{(i)} & \text{if } \mathrm{Re}\{p_n^{(i)}\} < 0, \\ -\mathrm{Re}\{p_n^{(i)}\} + \jmath\mathrm{Im}\{p_n^{(i)}\} & \text{if } \mathrm{Re}\{p_n^{(i)}\} > 0, \end{cases} \tag{8.71}$$

for $n = 1, \ldots, \bar{n}_r + \bar{n}_c$. We can see that, if a pole $p_n^{(i)}$ is unstable, the sign of its real part is inverted. Since in the tentative final fitting in step 8 of Algorithm 8.1 poles are fixed, condition (8.71) ensures the stability of the final model.

For frequency domain analyses, one may think that (8.70) is not necessary, since stability is not an issue. However, one can show that (8.70), in the frequency domain, becomes a condition for causality [82]. Causality means that the system will react to an excitation only after it has been applied, and not before. In other words, if the system input $u(t)$ begins at $t = t_0$ ($u(t) = 0$ for $t < t_0$), the system output will start varying only at or after $t = t_0$. All systems in nature are obviously causal, since they cannot "anticipate" the application of an excitation. Enforcing (8.70) ensures that VF model (8.58) is causal. If this is not the case, frequency domain analyses will succeed, but results may be inaccurate and unphysical. In particular, the VF model may underestimate the delay between input and output which is present in the real system, which may be important in some applications, such as the timing analysis of digital circuits. A complete discussion of causality is beyond the scope of this chapter, and the reader is referred to [82].

Overall, condition (8.70) simultaneously enforces the stability and causality of the VF model. This condition can be enforced without any accuracy penalty when the given samples H_k are error free, and thus faithfully represent the response of a causal and stable system. When samples are corrupted by noise or measurement errors, VF may be unable to reduce fitting error (8.6) to the desired level if condition (8.70) is enforced. This happens when the noise or errors in samples H_k are not causal functions themselves, and thus cannot be approximated with stable and causal poles [82]. Numerical algorithms exist to verify if the given samples H_k satisfy the causality condition required by VF to fit them with high accuracy [77, 78, 51, 76, 7].

In addition to causality and stability, passivity is another important property that one may want to impose on the VF model (8.58). This property characterizes those physical systems that are unable to generate energy on their own, simply due to the lack of energy sources or gain mechanisms inside them. A circuit made by positive resistors, capacitors and inductors is an example of a passive system, in contrast to an amplifying circuit. When applied to the response of a passive system, VF may still produce a non-passive model, due to approximation and numerical errors. However, passivity can be enforced a-posteriori, with the methods presented in Chapter 5 of this volume.

8.3.11 Numerical implementation

Vector Fitting is easy to implement, and several free codes are available [75, 38]. This section briefly describes a few changes to the basic templates in Algorithms 8.1 and 8.2 that can lead to a more robust and efficient implementation.

8.3.11.1 Order estimation

The VF templates in Algorithms 8.1 and 8.2 require the desired model order $\bar{n}$ as input. Typically, this is not known a priori, but can be determined during the fitting process using the VF algorithm with adding and skimming [33], as shown in the example of Section 8.3.7. In this method, an initial estimate of $\bar{n}$ is derived from the phase of the given samples H_k, and used in the first VF iteration. Then $\bar{n}$ is automatically increased or decreased based on the achieved error, as visible in Figure 8.8. If error e is still too high, the order is increased until either VF converges or it becomes evident that no further error reduction can be achieved, as in the last four iterations in Figure 8.8. Conversely, when the algorithm detects that some partial fractions in (8.58) give a negligible contribution over the frequency range of interest, the order $\bar{n}$ is reduced at the next iteration by removing such terms. This happens in the 13th iteration of the example in Section 8.3.7, where the order is reduced from 226 to 200.

8.3.11.2 Relaxed VF: a better normalization of the weighting function

In the original VF algorithm, the coefficients of weighting function (8.15) are normalized such that $w^{(i)}(\jmath\omega) \to 1$ when $\omega \to \infty$. It can be shown that this normalization is not optimal, and can slow down VF convergence when samples H_k are contaminated by noise. The relaxed VF algorithm [41] mitigates this issue by redefining the weighting function as

$$w^{(i)}(s) = w_0^{(i)} + \sum_{n=1}^{\bar{n}} \frac{w_n^{(i)}}{s - p_n^{(i-1)}}, \tag{8.72}$$

where $w_0^{(i)}$ is now free to depart from one. With this change, the fitting equation (8.37) becomes

$$R_{0,qm}^{(i)} + \sum_{n=1}^{\bar{n}} \frac{R_{n,qm}^{(i)}}{\jmath\omega_k - p_n^{(i-1)}} - H_{k,qm}\left(w_0^{(i)} + \sum_{n=1}^{\bar{n}} \frac{w_n^{(i)}}{\jmath\omega_k - p_n^{(i-1)}} \right) = 0. \tag{8.73}$$

Since (8.73) admits a trivial solution ($R_{n,qm}^{(i)} = w_n^{(i)} = 0\ \forall n$), the relaxed VF algorithm adds an additional constraint to exclude it [41]

$$\frac{1}{\bar{k}} \sum_{k=1}^{\bar{k}} \mathrm{Re}\left\{ w_0^{(i)} + \sum_{n=1}^{\bar{n}} \frac{w_n^{(i)}}{\jmath\omega_k - p_n^{(i-1)}} \right\} = 1. \tag{8.74}$$

This constraint can be seen as a more relaxed normalization of the weighting function. Equations (8.73) and (8.74) are then jointly solved in the least-squares sense. In the single-input single-output case ($\bar{q} = \bar{m} = 1$), the system to be solved takes the form

$$\begin{bmatrix} \Phi_0^{(i)} & -D_H \Phi_0^{(i)} \\ 0 & \frac{\beta}{\bar{k}} (1_{\bar{k}})^T \Phi_0^{(i)} \end{bmatrix} \begin{bmatrix} c_H^{(i)} \\ c_w^{(i)} \end{bmatrix} = \begin{bmatrix} 0 \\ \beta \end{bmatrix} \tag{8.75}$$

where

$$c_w^{(i)} = \begin{bmatrix} w_0^{(i)} & \dots & w_{\bar{n}}^{(i)} \end{bmatrix}^T. \tag{8.76}$$

In (8.75), β is a suitable weight to the last equation, which is typically set to [41]

$$\beta = \sqrt{\sum_{k=1}^{\bar{k}} |H_k|^2}. \tag{8.77}$$

8.4 Generalized and advanced VF algorithms

Since its inception in 1996, VF has inspired a generation of algorithms for the data-driven modeling of linear systems. These extensions either improve the original VF formulation, or extend it to different modeling scenarios. We briefly summarize the most relevant work in this area, and provide several bibliographic references where more details can be found.

8.4.1 Time domain VF algorithms

The original VF algorithm works in the frequency domain, and creates the reduced model from samples of the system frequency response. In some applications, however, it is more convenient to characterize the system in the time domain. For example, one may have simultaneous measurements of the system input $u(t_l)$ and output $y(t_l)$ at several time points t_l for $l = 1, \dots, \bar{l}$, as in the example of Section 8.3.4. In this scenario, one has two options. The first is to estimate the systems' frequency response from the time domain samples with the discrete Fourier transform, and apply VF in the frequency domain. However, the accuracy of the discrete Fourier transform depends significantly on the sampling rate of the given samples, and on their behavior near the boundaries $t = t_1$ and $t = t_{\bar{l}}$ of the acquisition window. These issues, if not well understood and managed, can result in an inaccurate time–frequency conversion, and degrade model quality.

The second option is to use the time domain VF algorithm [30, 31], which directly extracts (8.58) from the time domain samples $u(t_l)$ and $y(t_l)$. This is achieved by rewriting the fitting error (8.17) in the time domain, where multiplication by partial fraction $1/(s - p_n)$ becomes a convolution between $e^{p_n t}$ and the input or output samples. These convolutions can be computed by numerical integrations, leading to a time domain version of the original VF algorithm which closely follows the steps of the original frequency domain VF algorithm [35].

The time domain VF algorithm leads to a model in the continuous time domain. Alternatively, if the sampling period $\Delta t = t_{l+1} - t_l$ is constant, one can also apply the z-domain VF [59], which relies on the z transform as opposed to the Laplace transform.

This latter algorithm leads to a model in the discrete time domain, which can be expressed as a digital filter or as a set of difference equations (as opposed to differential equations).

8.4.2 Improved Vector Fitting formulations

In the QuadVF algorithm [23], a quadrature rule inspired by the H_2 error measure is used in conjunction with a suitable choice of frequency sampling points to improve the fidelity of the reduced model to the given samples. The same work also shows how one can incorporate derivative information, making QuadVF able to minimize a discrete Sobolev norm. In [24], this approach is extended to the multi-input multi-output case, and a way to control the McMillan degree[1] of the approximation is proposed, which helps to achieve smaller reduced models when $\bar{q}$ and $\bar{m}$ are high.

The numerical robustness of VF, which is already quite remarkable in its original formulation, is further improved in the Orthonormal VF algorithm [21]. This algorithm replaces partial fractions $1/(s-p_n)$ in (8.15) and (8.16) with orthonormal rational functions, achieving better numerical conditioning of the linear system (8.38) to be solved.

Another subject that received considerable attention is the robustness of VF against noise in the given samples H_k. Noise may arise from the measurement process or, if samples were obtained with a numerical simulation, from round-off errors, approximations, and convergence issues. The relaxed normalization discussed in Section 8.3.11.2 improves VF convergence in the presence of noise [41]. Furthermore, the VF with adding and skimming includes a mechanism to detect spurious poles caused by noise [33]. Since spurious poles impair VF convergence, they must be removed throughout iterations [33]. This mechanism is coupled with a robust way to adaptively refine model order $\bar{n}$ to maximize accuracy even when noise is significant [33]. Taking into account noise variance in the definition of the VF fitting error was also shown to improve convergence [26]. Finally, instrumental variables can be used to unbias the VF process from the effects of noise, leading to better accuracy and convergence at no additional cost [10].

8.4.3 VF algorithms for distributed systems

The efficient modeling of distributed systems is an open problem in model order reduction. A system is distributed when the time a signal takes to propagate from an input to an output is not negligible. In systems described by a Helmholtz (wave) equation, this happens when the physical size of the system is not negligible compared to the

1 The McMillan degree [93] of a matrix transfer function $H(s)$ is the order of a minimal state space realization of $H(s)$, such as the order N of the Gilbert realization discussed in Section 8.3.9.

wavelength. Propagation delays lead to the presence of irrational terms in the transfer function of the underlying system. Typically, these terms are in the form $e^{-s\tau}$ where τ is the propagation delay. Rational functions, including the partial fractions in (8.58) can accurately fit these irrational terms, up to arbitrary accuracy. However, if τ is not negligible, the required order may be large, and will quickly increase as τ grows. This leads to a large model which may burden subsequent simulations.

To overcome this issue, the core idea is to explicitly include exponential terms $e^{-s\tau_l}$ in the reduced model which will be fitted to the given samples. A popular choice is to define each element $\widetilde{H}_{qm}(s)$ of the model transfer function as

$$\sum_{l=1}^{\bar{l}} \left(r_{0,l} + \sum_{n=1}^{\bar{n}_l} \frac{r_{n,l}}{s - p_{n,l}} \right) e^{-s\tau_l}, \tag{8.78}$$

where the $_{qm}$ subscript was omitted from all coefficients for clarity. The exponential factors in (8.78) are meant to efficiently capture long propagation delays, while the rational terms between brackets will resolve the residual behavior of the system. Typically, since long propagation delays are already accounted for by the exponential terms, the order $\bar{n}_l$ of the rational factors can be kept quite low.

For systems with uniform cross-section along the direction of propagation, such as electrical transmission lines and fluid pipes, VF is used in conjunction to the method of characteristics to obtain an efficient distributed model [50, 2, 36, 61]. For distributed systems of general shape, several VF algorithms with delay terms have been proposed [15, 16, 13, 79, 67, 58]. In these algorithms, the first step is to identify the values of the relevant propagation delays τ_i present in the system. Given only frequency samples H_k, this is not a trivial task, and the dominant approach is to exploit time–frequency decompositions [39, 32, 67, 48]. Next, the coefficients of the remaining rational factors in the model are determined with a VF-like iterative process [16, 13, 79, 67, 58].

8.4.4 Parametric VF algorithms

The design process of an engineering system typically requires a large number of simulations for different values of design parameters, such as material properties, geometrical dimensions and operating conditions (e. g. bias voltages, temperature, ...). In early design stages, parametric simulations are used to explore the design space. Later on, they may be used to optimize design in order to meet specifications or improve performance. Moreover, parametric simulations also help designers to account for manufacturing variability during design. In the context of parametric simulations, conventional VF models may be inefficient. Indeed, every time a parameter changes, a new set of samples H_k must be obtained, and the fitting process has to be repeated from scratch.

A better solution is to create a parametric VF model which captures the system response with respect to both frequency s and some parameters of interest $\mu^{(1)}, \mu^{(2)}, \ldots$. The core idea behind parametric VF techniques [83, 80, 64, 20, 34] is to let residues R_n and poles p_n in (8.58) be parameter-dependent functions, such as polynomials in $\mu^{(1)}, \mu^{(2)}, \ldots$. Their coefficients can be determined with an iterative process analogous to the Sanathanan–Koerner iteration in Section 8.2, starting from samples of the system's frequency response obtained for multiple values of parameters $\mu^{(1)}, \mu^{(2)}, \ldots$. The main advantage of a parametric model is that, once generated, it can be reused many times for different parameter values within its range of validity. One of the challenges in the generation of parametric VF models is how to guarantee that the model will be stable and passive over the desired parameter range [81, 85, 25]. Recently, systematic solutions to this challenging problem have been proposed [92].

8.5 Conclusion

This chapter introduced the Vector Fitting algorithm, which has become one of the most popular tools for the extraction of linear reduced order models from samples of their response, collected in the frequency or in the time domain. Vector Fitting produces a rational model which approximately minimizes the least-squares error between the given samples and the model response. Determining model coefficients is originally a nonlinear least-squares problem, whose solution is prone to the typical issues of nonlinear minimization: high computational cost and problematic convergence due to local minima. Vector Fitting overcomes these issues by iteratively minimizing a linearization of the original problem, leveraging well-established methods for the solution of linear least-squares problems. Several strategies to obtain a robust and efficient implementation of VF have been reviewed. When properly implemented, Vector Fitting enjoys remarkable robustness, efficiency and versatility, typically converging in a handful of iterations. Finally, we reviewed the most prominent extensions of the original algorithm which have been proposed for data-driven modeling of time domain systems, noisy samples, distributed systems, and parametric systems.

Vector Fitting's superior performance and reliability lead to a widespread use in many different fields. Originally conceived to predict how transients propagate throughout power distribution networks, VF is the method of choice for the wideband modeling of overhead lines, underground cables and power transformers [61, 40, 62, 4, 35]. In electronic engineering, VF is extensively used to model the propagation of high-speed signals through interconnect networks found at the chip, package and printed circuit board level. These models are crucial for system design, and greatly help in preventing signal integrity, power integrity and electromagnetic compatibility issues [2, 68, 55, 74, 64, 1, 90]. The impact of VF in this area is confirmed by the fact that

all leading commercial tools for the design of high-frequency electronic circuits include a VF module. Applications in microwave engineering [56, 84, 19, 18] and digital filter design [89] have also been reported. Within computational electromagnetism, VF can be used to efficiently model the Green function of layered media, which is necessary to solve Maxwell's equations with integral equation methods [49, 11, 65]. The ability of VF to generate models compatible with transient simulations has also been exploited in the finite difference time domain (FDTD) method [57, 60], the finite element time domain method [12, 87], and the discontinuous Galerkin method [91]. Beyond electrical engineering, VF found countless applications in various domains, including acoustics [17, 66], fluid dynamics [3, 45, 35], mechanical engineering [35, 6], and in the thermal modeling of chemical batteries [44]. For a collection of VF applications and additional references, the reader is referred to [35].

Bibliography

[1] A. Chinea, S. Grivet-Talocia, H. Hu, P. Triverio, D. Kaller, C. Siviero, and M. Kindscher. Signal integrity verification of multi-chip links using passive channel macromodels. *IEEE Trans. Compon. Packag. Manuf. Technol.*, 1(6):920–933, 2011.

[2] R. Achar and M. S. Nakhla. Simulation of high-speed interconnects. *Proc. IEEE*, 89(5):693–728, 2001.

[3] A. Almondo and M. Sorli. Time domain fluid transmission line modelling using a passivity preserving rational approximation of the frequency dependent transfer matrix. *Int. J. Fluid Power*, 7(1):41–50, 2006.

[4] U. Annakkage, N.-K. C. Nair, Y. Liang, A. Gole, V. Dinavahi, B. Gustavsen, T. Noda, H. Ghasemi, A. Monti, M. Matar, and et al.. Dynamic system equivalents: A survey of available techniques. *IEEE Trans. Power Deliv.*, 27(1):411–420, 2012.

[5] G. Antonini. SPICE equivalent circuits of frequency-domain responses. *IEEE Trans. Electromagn. Compat.*, 45(3):502–512, 2003.

[6] E. Balmès. GARTEUR Group on Ground Vibration Testing. Results from the Test of a Single Structure by 12 Laboratories in Europe. In *15th International Modal Analysis Conference*, volume 3089, page 1346, 1997.

[7] L. L. Barannyk, H. A. Aboutaleb, A. Elshabini, and F. D. Barlow. Spectrally accurate causality enforcement using svd-based fourier continuations for high-speed digital interconnects. *IEEE Trans. Compon. Packag. Manuf. Technol.*, 5(7):991–1005, 2015.

[8] U. Baur, P. Benner, and L. Feng. Model order reduction for linear and nonlinear systems: a system-theoretic perspective. *Arch. Comput. Methods Eng.*, 21(4):331–358, 2014.

[9] D. S. Bayard. High-order multivariable transfer function curve fitting: Algorithms, sparse matrix methods and experimental results. *Automatica*, 30(9):1439–1444, 1994.

[10] A. Beygi and A. Dounavis. An instrumental variable vector-fitting approach for noisy frequency responses. *IEEE Trans. Microw. Theory Tech.*, 60(9):2702–2712, 2012.

[11] R. R. Boix, F. Mesa, and F. Medina. Application of total least squares to the derivation of closed-form Green's functions for planar layered media. *IEEE Trans. Microw. Theory Tech.*, 55(2):268–280, 2007.

[12] Y. Cai and C. Mias. Faster 3D finite element time domain-floquet absorbing boundary condition modelling using recursive convolution and vector fitting. *IET Microw. Antennas Propag.*, 3(2):310–324, 2009.

[13] A. Charest, M. S. Nakhla, R. Achar, D. Saraswat, N. Soveiko, and I. Erdin. Time domain delay extraction-based macromodeling algorithm for long-delay networks. *IEEE Trans. Adv. Packaging*, 33(1):219–235, 2010.
[14] A. Chinea and S. Grivet-Talocia. On the parallelization of vector fitting algorithms. *IEEE Trans. Compon. Packag. Manuf. Technol.*, 1(11):1761–1773, 2011.
[15] A. Chinea, P. Triverio, and S. Grivet-Talocia. Compact macromodeling of electrically long interconnects. In *Proc. of the 17th Topical Meeting on Electrical Performance of Electronic Packaging (EPEP 2008)*, pages 199–202. IEEE, 2008.
[16] A. Chinea, P. Triverio, and S. Grivet-Talocia. Delay-based macromodeling of long interconnects from frequency-domain terminal responses. *IEEE Trans. Adv. Packaging*, 33(1):246–256, 2010.
[17] B. Cotté, P. Blanc-Benon, C. Bogey, and F. Poisson. Time-domain impedance boundary conditions for simulations of outdoor sound propagation. *AIAA J.*, 47(10):2391–2403, 2009.
[18] D. De Jonghe and G. Gielen. Characterization of analog circuits using transfer function trajectories. *IEEE Trans. Circuits Syst. I, Regul. Pap.*, 59(8):1796–1804, 2012.
[19] D. Deschrijver, G. Avolio, D. Schreurs, T. Dhaene, G. Crupi, and L. Knockaert. Microwave small-signal modelling of FinFETs using multi-parameter rational fitting method. *Electron. Lett.*, 47(19):1084–1086, 2011.
[20] D. Deschrijver, T. Dhaene, and D. De Zutter. Robust parametric macromodeling using multivariate orthonormal vector fitting. *IEEE Trans. Microw. Theory Tech.*, 56(7):1661–1667, 2008.
[21] D. Deschrijver, B. Haegeman, and T. Dhaene. Orthonormal vector fitting: A robust macromodeling tool for rational approximation of frequency domain responses. *IEEE Trans. Adv. Packaging*, 30(2):216–225, 2007.
[22] D. Deschrijver, M. Mrozowski, T. Dhaene, and D. De Zutter. Macromodeling of multiport systems using a fast implementation of the vector fitting method. *IEEE Microw. Wirel. Compon. Lett.*, 18(6):383–385, 2008.
[23] Z. Drmac, S. Gugercin, and C. Beattie. Quadrature-based vector fitting for discretized h_2 approximation. *SIAM J. Sci. Comput.*, 37(2):A625–A652, 2015.
[24] Z. Drmac, S. Gugercin, and C. Beattie. Vector fitting for matrix-valued rational approximation. *SIAM J. Sci. Comput.*, 37(5):A2346–A2379, 2015.
[25] F. Ferranti, L. Knockaert, and T. Dhaene. Guaranteed passive parameterized admittance-based macromodeling. *IEEE Trans. Adv. Packaging*, 33(3):623–629, 2010.
[26] F. Ferranti, Y. Rolain, L. Knockaert, and T. Dhaene. Variance weighted vector fitting for noisy frequency responses. *IEEE Microw. Wirel. Compon. Lett.*, 20(4):187–189, 2010.
[27] E. G. Gilbert. Controllability and observability in multivariable control systems. *J. Soc. Ind. Appl. Math., A, on Control*, 1(2):128–151, 1963.
[28] G. H. Golub and C. F. V. Loan. *Matrix Computations*. Johns Hopkins University Press, 1996.
[29] G. H. Golub and R. J. Plemmons. Large-scale geodetic least-squares adjustment by dissection and orthogonal decomposition. *Linear Algebra Appl.*, 34:3–28, 1980.
[30] S. Grivet-Talocia. Package macromodeling via time-domain vector fitting. *IEEE Microw. Wirel. Compon. Lett.*, 13(11):472–474, 2003.
[31] S. Grivet-Talocia. The time-domain vector fitting algorithm for linear macromodeling. *Int. J. Electron. Commun.*, 58(4):293, 2004.
[32] S. Grivet-Talocia. Delay-based macromodels for long interconnects via time–frequency decompositions. In *2006 IEEE Electrical Performance of Electronic Packaging*, pages 199–202. IEEE, 2006.
[33] S. Grivet-Talocia and M. Bandinu. Improving the convergence of vector fitting for equivalent circuit extraction from noisy frequency responses. *IEEE Trans. Electromagn. Compat.*, 48(1):104–120, 2006.

[34] S. Grivet-Talocia and E. Fevola. Compact parameterized black-box modeling via Fourier-rational approximations. *IEEE Trans. Electromagn. Compat.*, 59(4):1133–1142, 2017.
[35] S. Grivet-Talocia and B. Gustavsen. *Passive macromodeling: Theory and applications*. John Wiley & Sons, 2015.
[36] S. Grivet-Talocia, H. -M. Huang, A. E. Ruehli, F. Canavero, and I. Elfadel. Transient analysis of lossy transmission lines: An efficient approach based on the method of characteristics. *IEEE Trans. Adv. Packaging*, 27(1):45–56, 2004.
[37] S. Grivet-Talocia, S. Olivadese, and P. Triverio. A compression strategy for rational macromodeling of large interconnect structures. In *2011 IEEE Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS)*, pages 53–56. IEEE, 2011.
[38] B. Gustavsen. The Vector Fitting Website. https://www.sintef.no/projectweb/vectfit/. Accessed: 2018-12-06.
[39] B. Gustavsen. Time delay identification for transmission line modeling. In *8th IEEE Workshop on Signal Propagation on Interconnects*, pages 103–106. IEEE, 2004.
[40] B. Gustavsen. Wide band modeling of power transformers. *IEEE Trans. Power Deliv.*, 19(1):414–422, 2004.
[41] B. Gustavsen. Improving the pole relocating properties of vector fitting. *IEEE Trans. Power Deliv.*, 21(3):1587–1592, 2006.
[42] B. Gustavsen and A. Semlyen. Rational approximation of frequency domain responses by vector fitting. *IEEE Trans. Power Deliv.*, 14(3):1052–1061, 1999.
[43] W. Hendrickx and T. Dhaene. A discussion of "Rational approximation of frequency domain responses by vector fitting". *IEEE Trans. Power Syst.*, 21(1):441–443, 2006.
[44] X. Hu, L. Chaudhari, S. Lin, S. Stanton, S. Asgari, and W. Lian. A state space thermal model for HEV/EV battery using vector fitting. In *2012 IEEE Transportation Electrification Conference and Expo (ITEC)*, pages 1–8. IEEE, 2012.
[45] S. Jaensch, C. Sovardi, and W. Polifke. On the robust, flexible and consistent implementation of time domain impedance boundary conditions for compressible flow simulations. *J. Comput. Phys.*, 314:145–159, 2016.
[46] M. T. Kassis, M. Kabir, Y. Q. Xiao, and R. Khazaka. Passive reduced order macromodeling based on loewner matrix interpolation. *IEEE Trans. Microw. Theory Tech.*, 64(8):2423–2432, 2016.
[47] L. Knockaert. Comments on "macromodeling of multiport systems using a fast implementation of the vector fitting method". *IEEE Microw. Wirel. Compon. Lett.*, 19(9):602, 2009.
[48] I. Kocar and J. Mahseredjian. New procedure for computation of time delays in propagation function fitting for transient modeling of cables. *IEEE Trans. Power Deliv.*, 31(2):613–621, 2016.
[49] V. N. Kourkoulos and A. C. Cangellaris. Accurate approximation of Green's functions in planar stratified media in terms of a finite sum of spherical and cylindrical waves. *IEEE Trans. Antennas Propag.*, 54(5):1568–1576, 2006.
[50] D. B. Kuznetsov and J. E. Schutt-Ainé. Optimal transient simulation of transmission lines. *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, 43(2):110–121, 1996.
[51] S. Lalgudi. On checking causality of tabulated *S*-parameters. *IEEE Trans. Compon. Packag. Manuf. Technol.*, 3(7):1204–1217, 2013.
[52] S. Lefteriu and A. C. Antoulas. A new approach to modeling multiport systems from frequency-domain data. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 29(1):14–27, 2010.
[53] S. Lefteriu and A. C. Antoulas. On the convergence of the vector-fitting algorithm. *IEEE Trans. Microw. Theory Tech.*, 61(4):1435–1443, 2013.
[54] E. Levy. Complex-curve fitting. *IRE Trans. Autom. Control*, AC–4(1):37–43, 1959.
[55] E. -P. Li, X. -C. Wei, A. C. Cangellaris, E. -X. Liu, Y. -J. Zhang, M. D'amore, J. Kim, and T. Sudo. Progress review of electromagnetic compatibility analysis technologies for packages, printed circuit boards, and novel interconnects. *IEEE Trans. Electromagn. Compat.*, 52(2):248–265, 2010.

[56] C. -K. Liao, C. -Y. Chang, and J. Lin. A vector-fitting formulation for parameter extraction of lossy microwave filters. *IEEE Microw. Wirel. Compon. Lett.*, 17(4):277–279, 2007.
[57] H. Lin, M. F. Pantoja, L. D. Angulo, J. Alvarez, R. G. Martin, and S. G. Garcia. FDTD modeling of graphene devices using complex conjugate dispersion material model. *IEEE Microw. Wirel. Compon. Lett.*, 22(12):612–614, 2012.
[58] M. Luo and K. -M. Huang. An extended delay-rational macromodel for electromagnetic interference analysis of mixed signal circuits. *Prog. Electromagn. Res.*, 127:189–210, 2012.
[59] Y. S. Mekonnen and J. E. Schutt-Aine. Broadband macromodeling of sampled frequency data using z-domain vector-fitting method. In *2007 IEEE Workshop on Signal Propagation on Interconnects*, pages 45–48. IEEE, 2007.
[60] K. A. Michalski. On the low-order partial-fraction fitting of dielectric functions at optical wavelengths. *IEEE Trans. Antennas Propag.*, 61(12):6128–6135, 2013.
[61] A. Morched, B. Gustavsen, and M. Tartibi. A universal model for accurate calculation of electromagnetic transients on overhead lines and underground cables. *IEEE Trans. Power Deliv.*, 14(3):1032–1038, 1999.
[62] T. Noda. Identification of a multiphase network equivalent for electromagnetic transient calculations using partitioned frequency response. *IEEE Trans. Power Deliv.*, 20(2):1134–1142, 2005.
[63] S. B. Olivadese and S. Grivet-Talocia. Compressed passive macromodeling. *IEEE Trans. Compon. Packag. Manuf. Technol.*, 2(8):1378–1388, 2012.
[64] P. Triverio, S. Grivet-Talocia, M. Bandinu, and F. Canavero. Geometrically-parameterized circuit models of printed circuit board traces inclusive of antenna coupling. *IEEE Trans. Electromagn. Compat.*, 52:471–478, 2010.
[65] A. G. Polimeridis, T. V. Yioultsis, and T. D. Tsiboukis. A robust method for the computation of Green's functions in stratified media. *IEEE Trans. Antennas Propag.*, 55(7):1963–1969, 2007.
[66] S. R. Robinson, C. T. Nguyen, and J. B. Allen. Characterizing the ear canal acoustic impedance and reflectance by pole-zero fitting. *Hear. Res.*, 301:168–182, 2013.
[67] S. Roy and A. Dounavis. Transient simulation of distributed networks using delay extraction based numerical convolution. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 30(3):364–373, 2011.
[68] A. E. Ruehli and A. C. Cangellaris. Progress in the methodologies for the electrical modeling of interconnects and electronic packages. *Proc. IEEE*, 89(5):740–771, 2001.
[69] C. Sanathanan and J. Koerner. Transfer function synthesis as a ratio of two complex polynomials. *IEEE Trans. Autom. Control*, 8(1):56–58, 1963.
[70] W. H. Schilders. The need for novel model order reduction techniques in the electronics industry. In *Model reduction for circuit simulation*, pages 3–23. Springer, 2011.
[71] M. K. Sharp, G. M. Pantalos, L. Minich, L. Y. Tani, E. C. McGough, and J. A. Hawkins. Aortic input impedance in infants and children. *J. Appl. Physiol.*, 88(6):2227–2239, 2000.
[72] G. Shi. On the nonconvergence of the vector fitting algorithm. *IEEE Trans. Circuits Syst. II, Express Briefs*, 63(8):718–722, 2016.
[73] K. Steiglitz and L. McBride. A technique for the identification of linear systems. *IEEE Trans. Autom. Control*, 10(4):461–464, 1965.
[74] M. Swaminathan, D. Chung, S. Grivet-Talocia, K. Bharath, V. Laddha, and J. Xie. Designing and modeling for power integrity. *IEEE Trans. Electromagn. Compat.*, 52(2):288–310, 2010.
[75] P. Triverio. Vector Fitting Resources. http://www.modelics.org/vf.html. Accessed: 2019-08-23.
[76] P. Triverio. Robust causality check for sampled scattering parameters via a filtered fourier transform. *IEEE Microw. Wirel. Compon. Lett.*, 24(2):72–74, 2014.
[77] P. Triverio and S. Grivet-Talocia. A robust causality verification tool for tabulated frequency data. In *2006 IEEE Workship on Signal Propagation on Interconnects*, pages 65–68. IEEE, 2006.

[78] P. Triverio and S. Grivet-Talocia. Robust causality characterization via generalized dispersion relations. *IEEE Trans. Adv. Packaging*, 31(3):579–593, 2008.
[79] P. Triverio, S. Grivet-Talocia, and A. Chinea. Identification of highly efficient delay-rational macromodels of long interconnects from tabulated frequency data. *IEEE Trans. Microw. Theory Tech.*, 58(3):566–577, 2010.
[80] P. Triverio, S. Grivet-Talocia, and M. S. Nakhla. An improved fitting algorithm for parametric macromodeling from tabulated data. In *2008 12th IEEE Workshop on Signal Propagation on Interconnects*, pages 1–4. IEEE, 2008.
[81] P. Triverio, S. Grivet-Talocia, and M. S. Nakhla. A parameterized macromodeling strategy with uniform stability test. *IEEE Trans. Adv. Packaging*, 32(1):205–215, 2009.
[82] P. Triverio, S. Grivet-Talocia, M. S. Nakhla, F. G. Canavero, and R. Achar. Stability, causality, and passivity in electrical interconnect models. *IEEE Trans. Adv. Packaging*, 30(4):795–808, 2007.
[83] P. Triverio, M. Nakhla, and S. Grivet-Talocia. Parametric macromodeling of multiport networks from tabulated data. In *2007 IEEE Workshop on Electrical Performance of Electronic Packaging*, pages 51–54. IEEE, 2007.
[84] P. Triverio, M. Nakhla, and S. Grivet-Talocia. Extraction of parametric circuit models from scattering parameters of passive RF components. In *The 40th European Microwave Conference*, pages 1635–1638. IEEE, 2010.
[85] P. Triverio, M. S. Nakhla, and S. Grivet-Talocia. Passive parametric macromodeling from sampled frequency data. In *2010 IEEE 14th Workshop on Signal Propagation on Interconnects*, pages 117–120. IEEE, 2010.
[86] P. Verboven, P. Guillaume, and B. Cauberghe. Multivariable frequency–response curve fitting with application to modal parameter estimation. *Automatica*, 41(10):1773–1782, 2005.
[87] R. Wang and J. -M. Jin. Incorporation of multiport lumped networks into the hybrid time-domain finite-element analysis. *IEEE Trans. Microw. Theory Tech.*, 57(8):2030–2037, 2009.
[88] W. River. Technology. CMP-28 Channel Modeling Platform. https://wildrivertech.com/index.php/cmp-28-cmp-32. Accessed: 2019-05-17.
[89] N. Wong and C. -U. Lei. IIR approximation of FIR filters via discrete-time vector fitting. *IEEE Trans. Signal Process.*, 56(3):1296–1302, 2008.
[90] T. -L. Wu, F. Buesink, and F. Canavero. Overview of signal integrity and EMC design technologies on PCB: Fundamentals and latest progress. *IEEE Trans. Electromagn. Compat.*, 55(4):624–638, 2013.
[91] S. Yan, P. Wang, C. -Y. Tian, and L. Li. Analysis of graphene-based devices using wave equation-based discontinuous Galerkin time domain method. *IEEE Antennas Wirel. Propag. Lett.*, 17(12):2169–2173, 2018.
[92] A. Zanco, S. Grivet-Talocia, T. Bradde, and M. De Stefano. Enforcing passivity of parameterized LTI macromodels via Hamiltonian-driven multivariate adaptive sampling. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 39(4):225–238, 2020.
[93] K. Zhou, J. C. Doyle, and K. Glover. *Robust and optimal control*, volume 40. Prentice Hall, 1996.