

MFT-NNLS: A Toolbox For Passive Macromodeling From Large Immittance and Scattering Data Sets Using Vector Fitting And Residue Perturbation in Matlab

Bjørn Gustavsen, *Fellow, IEEE*

Abstract—This paper describes a Matlab toolbox (MFT-NNLS) for passive macromodeling of large-scale problems defined by admittance, impedance or scattering data in the frequency domain. Vector Fitting (VF) is used for calculating an initial rational model having a common pole set. Passivity can afterwards be enforced by residue perturbation of the associated pole-residue model. Highly efficient implementations are utilized, including the so-called fast implementation of VF, and passivity enforcement with problem compacting via the QR-NNLS method with usage of a fast LU-based non-negative least squares solver. The final result is a rational model satisfying the physical conditions of realness, stability, passivity and reciprocity. The rational model is returned on alternative forms, including a pole-residue model and a state-space model with real-valued or complex-valued parameters. Optional parameters are available for tuning the behavior of the software to the given application, e.g., least squares weighting schemes and alternative methods for passivity assessment. Model export to leading electromagnetic transient programs is included. Several applications are shown, including large-scale problems from high-voltage power systems (high orders, many terminals) and high-speed electronics (many ports). The software consists of a collection of Matlab functions.

Index Terms—vector fitting, residue perturbation, passivity enforcement, admittance, impedance, scattering, pole-residue model, state space model, macromodel, Matlab.

I. INTRODUCTION

DATA driven modeling of linear time invariant components and subsystems is conveniently achieved using rational function approximations of the component terminal or port behaviors [1]. Such approach is frequently based on measured or simulated frequency immittance or scattering frequency domain responses. The use of rational function approximation leads to a model on pole-residue or state-space form, enabling highly efficient time domain simulations [1]. The actual rational fitting was originally achieved by fitting a ratio of two polynomials to the data in the Sanathanan-Koerner iteration [2].

The later introduction of the pole relocating vector fitting (VF) method [3] and its many variants greatly improved the accuracy and robustness, making it widely applied in many disciplines. Improved convergence was achieved by a relaxed constraint in the pole-relocation step [4], and the computational efficiency was greatly improved for multi-terminal applications by utilizing decoupling via the sparsity pattern of the associated QR factorization problem [5]. The ever-growing problem sizes to be tackled have further led to even faster solvers for VF, by use of parallel computation on multi-core CPUs [6] and GPUs [7], [8]. Other VF variants focus on modal fitting [9] and load impedance-aware fitting [10], and fitting input-output response $\{\mathbf{u}(s), \mathbf{y}(s)\}$ data sets instead of impulse responses $\mathbf{H}(s)$ [11]. Several viable alternatives to VF have also appeared, e.g., Loewner tangential interpolation [12], rational Krylov method [13] and the Adaptive Antoulas-Anderson (AAA) algorithm [14].

In the case of immittance and scattering problems, it is necessary to enforce passivity for the model to guarantee stable time domain simulations. None of the aforementioned rational fitting methods are capable of enforcing model passivity by construction. Several methods have therefore been proposed which aim at enforcing the passivity as a postprocessing step whereby a perturbation is made to the original model [1]. The non-linear relation between the passivity constraint and the model's parameters necessitates the use of iterations, during which the presence of passivity violations are precisely quantified. A number of different passivity enforcement schemes have been proposed, but most methods are based on iteratively solving a constrained linear least squares (LS) problem to minimize the change to the original problem [15]–[22], with passivity assessment based on frequency sweeping [23] or on test matrices [24],[25]. The pole-residue model resulting from VF is here particularly useful because the constrained linear LS problem can be conveniently set up using the residue matrix elements as free variables, so-called residue perturbation (RP).

Manuscript received September 29, 2025,...

B. Gustavsen is with SINTEF Energy Research, N-7465 Trondheim, Norway (e-mail: bjorn.gustavsen@sintef.no).

This work was supported by Northwind (Norwegian Research Centre on Wind Energy), www.northwindresearch.no, project code 321954.

One reason for the wide use of VF is that it was made freely available on the internet soon after its development [26], in the form of Matlab scripts. Later, a collection of files in the “Matrix Fitting Toolbox” (MFT) was made available [27], [28] which aim at calculating a stable and passive model to immittance and scattering data sets, with use of VF and passivity enforcement by RP. Scripts for generating data files for model inclusion in simulation software (ATP, PSCAD) were also included. Although a viable tool for data driven modeling, it has certain limitations. In particular, the passivity assessment and enforcement are limited to small and medium-scale problems as the solving involves large system matrices, and the robustness of the passivity iterations can be poor in some cases due to the adopted solver. The need for large-scale modeling capability is becoming a major issue when developing frequency-dependent network equivalents (FDNEs), permitting network models to be shared without revealing internal network details [29]. Here, the widespread use of (non-linear) power electronic converters in the grid introduces additional FDNE terminals, resulting in a larger, more challenging fitting problem. Frequency-dependent white-box modeling of power transformers poses a similar problem as the dimension of the branch impedance matrix to be fitted can easily exceed one hundred [30].

This paper describes a new implementation (MFT-NNLS) which overcomes the limitations of the original MFT, making it more suitable for handling large-scale problems. The VF implementation still makes use of relaxation [4] for improved convergence, and the decoupling approach in [5] for faster solving. Computational efficiency is further improved by calculating a more suitable pole set before fitting the full matrix $\mathbf{H}(s)$. The pole set is now obtained by fitting the diagonal elements of $\mathbf{H}(s)$ instead of the matrix trace as used in the original MFT implementation. Also, a number of control parameters are available, for instance to enforce asymptotic passive behavior. The passivity enforcement step is still based on residue perturbation while minimizing the change to the model’s behavior in the LS sense. The computational efficiency is significantly improved with use of a compacted cost function with constraints [19], [20] that is efficiently solved for using a fast non-negative least squares solver [31]. The approach is applicable for both immittance and scattering problems. A number of user control features are available, e.g., control of least squares (LS) weighting and number of free variables. The user can also choose between passivity assessment using test matrices or frequency sweeping. The code is downloadable from [26].

The paper is organized as follows. Section II gives an overview of the computational procedure and main features of MFT. Section III describes the functionality and user interaction of the *VFdriver* routine, which calculates an initial, stable model using VF. A similar description is provided in Section IV for the *RPdriver* routine, which enforces passivity of the model from *VFdriver*, by use of a highly efficient residue perturbation scheme. Section V describes available control settings of MFT, and how it can be used for reducing calculation time. Section VI gives an overview of four test cases, whose

application results are detailed in Section VII (transformer black-box modeling), Section VIII (frequency-dependent network equivalent (FDNE) modeling), Section IX (transformer white-box modeling), and Section X (high-speed interconnect modeling). The paper ends with a discussion in Section XII and a conclusion in Section XIII.

II. OVERVIEW OF CALCULATION PROCEDURE

The program fits a rational model to a set of admittance $\mathbf{Y}(s)$, impedance $\mathbf{Z}(s)$ or scattering data $\mathbf{S}(s)$, given as discrete function of frequency $s = j\omega$. The resulting rational model is given on pole-residue form (1) and on state-space form (2), where $\mathbf{H} = \mathbf{H}^T$ represents the data (\mathbf{Y} , \mathbf{Z} or \mathbf{S}).

$$\mathbf{H}^{P \times P}(s) = s\mathbf{R}_{-1}^{P \times P} + \mathbf{R}_0^{P \times P} + \sum_{i=1}^N \frac{\mathbf{R}_i^{P \times P}}{s - a_i} \quad (1)$$

$$\mathbf{H}^{P \times P}(s) = s\mathbf{E}^{P \times P} + \mathbf{D}^{P \times P} + \mathbf{C}^{P \times PN} (s\mathbf{I}^{PN \times PN} - \mathbf{A}^{PN \times PN})^{-1} \mathbf{B}^{PN \times P} \quad (2)$$

In (1) and (2), model parameters $\mathbf{E} = \mathbf{R}_{-1}$ and $\mathbf{D} = \mathbf{R}_0$ are optional parameters. In the case of scattering data \mathbf{S} , the usual assumption is made that $\mathbf{E} = \mathbf{R}_{-1} = \mathbf{0}$. In the following the term \mathbf{R}_{-1} will be dropped to simplify the description.

The rational fitting is achieved using Vector Fitting (VF), giving a model compliant with the physicality requirement of model parameters being real-valued or complex conjugate, and the poles being constrained to the negative half-plane. Additionally, the user can request that the model also satisfies the constraint of passivity, via residue perturbation (RP) of the model created by VF. The two steps have been implemented in two separate functions, *VFdriver* and *RPdriver*.

The final model can be exported for use with the mainstream simulation programs, EMTP, PSCAD and ATP, with use of routines *netgen_EMTP*, *netgen_PSCAD*, and *netgen_ATP*.

III. VFDRIVER

A. Method

The computational flow of *VFdriver* is shown in Fig. 1. *VFdriver* places the diagonal elements of \mathbf{H} in a vector \mathbf{h}_1 of frequency responses and applies least squares (LS) fitting of the data with a rational model (3) using VF [3], with use of the relaxed implementation [4] for improved convergence and accuracy, and the fast implementation [5] for improved computational efficiency. The result is a pole-residue model with common poles for all elements in \mathbf{h}_1 (3). During the iterative pole relocation, any pole in the right half plane is mirrored to the left half plane, thereby guaranteeing a model with stable poles.

$$\mathbf{h}_1(s) = \mathbf{r}_{1,0} + \sum_{i=1}^N \frac{\mathbf{r}_{1,i}}{s - a_{1,i}} \quad (3)$$

The obtained poles are reused as initial (improved) poles for

fitting the elements of a vector \mathbf{h}_2 which contains the upper triangular elements of \mathbf{H} . The same procedure with pole relocation and pole flipping is used also here.

Following the pole-relocating iterations, the elements of the residue vectors $\mathbf{r}_{2,i}$ are copied to the respective positions of the residue matrices \mathbf{R}_i of (1), which thereby result symmetric by construction. Finally, the pole-residue model (1) is converted into a state-space model (2). The state-space model has a diagonal state matrix \mathbf{A} and a (sparse) input matrix \mathbf{B} consisting of ones and zeros, being a selector matrix.

B. User Interaction

In addition to the essential input parameters (data set $\{s_k, \mathbf{H}(s_k)\}$ and model order N), the user controls the computations via a set of parameters, including:

- Number of pole relocating iterations (M_1) for fitting the diagonal elements $\mathbf{h}_1(s)$ of $\mathbf{H}(s)$ by a common pole set, for fast calculation of an improved pole set.
- Number of pole relocating iterations (M_2) for fitting the upper triangle elements $\mathbf{h}_2(s)$ of the full matrix $\mathbf{H}(s)$.
- Least squares frequency-dependent weighting options. Options include 1) user defined weighting for all matrix elements, and 2) parameter-controlled weighting schemes as described in Table I.
- Conversion of state-space model from diagonal form with complex matrices to form with real-valued matrices.
- Asymptotic model passivity by enforcing $\mathbf{R}_0 = \mathbf{D}$ to comply with passivity constraints, $\lambda_j > tol$ (immittance) or $\sigma_j < 1 - tol$ (scattering), $j = 1, 2, \dots, P$.
- Option for removing high-frequency out-of-band poles, in order to mitigate high-frequency passivity violations.

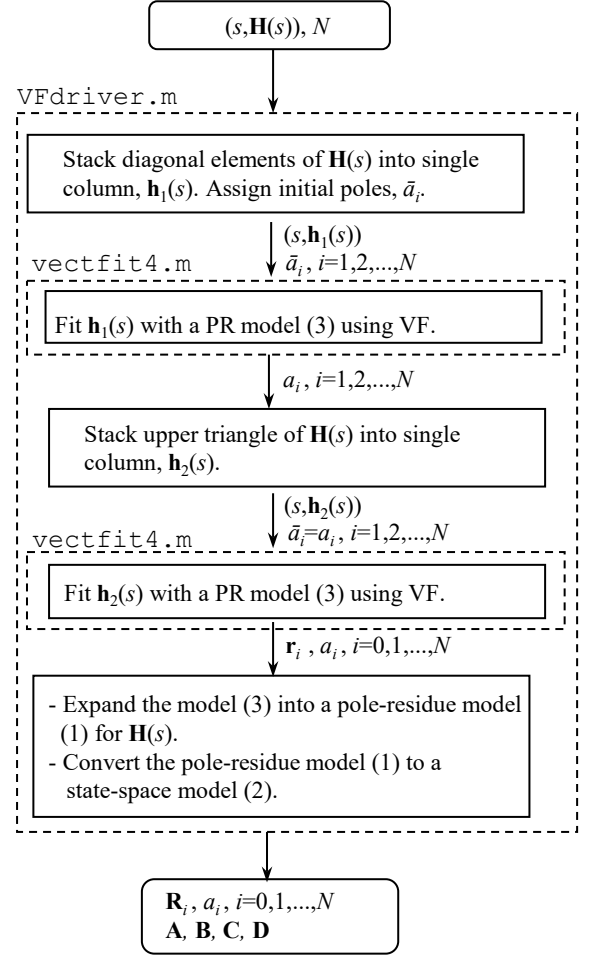


Fig. 1. Calculating a rational model using *VFdriver*.

TABLE I
LEAST SQUARES FREQUENCY-DEPENDENT WEIGHTING OPTIONS FOR
VFDRIVER AND RPDRIVER.

OPTION	WEIGHTING SCHEME
1	Unitary (no weighting)
2	Inverse of element magnitude
3	Inverse of square-root of element magnitude
4	Inverse of matrix norm
5	Inverse of square-root of matrix norm

IV. RPDRIVER

A. Method

RPdriver enforces passivity for the model obtained by *VFdriver* by iterative perturbation of the residue matrices $\{\mathbf{R}_i, i = 1, 2, \dots, N\}$. The conditions for passivity are the following [1].

Admittance: All eigenvalues of the real part $\mathbf{G}(s) = \text{Re}\{\mathbf{Y}(s)\}$ are positive, for all frequencies,

$$\lambda_j(\mathbf{G}(s)) > 0 \quad \forall s, j = 1, 2, \dots, P \quad (4a)$$

Impedance: All eigenvalues of the real part $\mathbf{R}(s) = \text{Re}\{\mathbf{Z}(s)\}$ are positive for all frequencies,

$$\lambda_j(\mathbf{R}(s)) > 0 \quad \forall s, j = 1, 2, \dots, P \quad (4b)$$

Scattering: All singular values of $\mathbf{S}(s)$ are smaller than unity for all frequencies,

$$\sigma_j(\mathbf{S}(s)) < 1 \quad \forall s, j = 1, 2, \dots, P \quad (4c)$$

The presence of passivity violations is assessed by calculating crossover frequencies where there exists a passivity status change for an eigenvalue by (4a) (admittance), by (4b) (impedance), or for a singular value by (4c) (scattering). The crossover frequencies are used for establishing frequency bands of passivity violations whose maximum points of violations are identified by frequency sweeping of the eigenvalues (4a), (4b) or singular values (4c).

The crossover frequencies can be established in two alternative ways:

- 1) Via frequency samples provided by the user, or
- 2) Via a test matrix calculated from the original model.

Option 1) can be very fast, but it is not reliable as it depends on the sample density and placements. Option 2) is reliable but can be time-consuming for large-scale problems. In Section X.B it is shown how the CPU time can be reduced by combining the two approaches.

In the case of Option 2), crossover frequencies ω_* are calculated based on the eigenvalues of a suitable test matrix. For the immittance (admittance or impedance) case, crossover frequencies are given as the square root of the subset of positive real eigenvalues ω_*^2 of $\mathbf{P}_{Y,Z}$ (5a) [24]. In the scattering case, the crossover frequencies $j\omega_*$ are given by the square root of the subset of negative real eigenvalues $-\omega_*^2$ of \mathbf{P}_S (5b) [25].

$$\mathbf{P}_{Y,Z} = \mathbf{A}(\mathbf{B}\mathbf{D}^{-1}\mathbf{C} - \mathbf{A}) \quad (5a)$$

$$\mathbf{P}_S = (\mathbf{A} - \mathbf{B}(\mathbf{D} - \mathbf{I})^{-1}\mathbf{C})(\mathbf{A} - \mathbf{B}(\mathbf{D} + \mathbf{I})^{-1}\mathbf{C}) \quad (5b)$$

The perturbation is made such that the relevant passivity condition (4a), (4b) or (4c) is fulfilled while minimizing the LS change to the model's behavior.

The perturbation makes use of first order approximations [34] for the relation between matrix elements and eigenvalues (6a) for immittance problems, and between matrix elements and singular values (6b) for scattering problems,

$$\Delta\lambda_j \approx \frac{\mathbf{t}_j^T (\Delta\mathbf{G}) \mathbf{t}_j}{\mathbf{t}_j^T \mathbf{t}_j}, j = 1 \dots P \quad (6a)$$

$$\Delta\sigma_j \approx \text{Re}\{\mathbf{u}_j^H \Delta\mathbf{S} \mathbf{v}_j\}, j = 1 \dots P \quad (6b)$$

where \mathbf{t}_j in (6a) is the j th eigenvector of the real-valued and symmetrical $\mathbf{G} = \text{Re}\{(\mathbf{Y} + \mathbf{Y}^T)\} / 2$. In the impedance case, \mathbf{G} is replaced with $\mathbf{R} = (\mathbf{Z} + \mathbf{Z}^T) / 2$. In (6b), \mathbf{u}_j and \mathbf{v}_j are the j th row and column of \mathbf{U} and \mathbf{V} of the singular value decomposition (7).

$$\mathbf{S}(\omega) = \mathbf{U}(\omega)\mathbf{\Sigma}(\omega)\mathbf{V}^H(\omega) \quad (7).$$

Introducing the residue matrix elements as free variables, the

following constrained linear least squares problem is established,

$$\min_{\mathbf{x}} \|\mathbf{A}\mathbf{x}\| \quad (8a)$$

$$\mathbf{C}\mathbf{x} > \mathbf{d} \quad (8b)$$

which attempts to find a model satisfying the linearized constraint condition (4a), (4b) or (4c) while minimizing the change to the model's behavior $\Delta\mathbf{Y}(s)$, $\Delta\mathbf{Z}(s)$ or $\Delta\mathbf{S}(s)$ over the user-provided frequency samples. The actual solving is performed by casting the model into a compacted form (9) via block-wise QR decomposition [19],

$$\min_{\mathbf{y}} \|\mathbf{y}\| \quad \text{s.t.} \quad \mathbf{C}\bar{\mathbf{R}}^{-1}\mathbf{y} > \mathbf{d} \quad (9)$$

Matrices \mathbf{E} and \mathbf{f} are established (10a) which permits the problem to be solved as the non-negative least squares (NNLS) problem (10b) [32]. From \mathbf{u} is recovered vector \mathbf{y} in (9) using (10c), from which the solution vector \mathbf{x} in (8a) is calculated using (10d), where $\boldsymbol{\varepsilon}$ is the NNLS residual (10e) and col_E is the number of columns in \mathbf{E} .

An active-set solver is employed which remains highly efficient also for large scale problems with many constraints (columns of \mathbf{E}), thanks to an LU-based implementation which avoids re-factorization by updating an existing LU factorization within the internal iteration loop [31]. The calculation of \mathbf{A} in (10b) and the following (block-wise) QR factorization is performed a single time only, before entering the perturbation loop.

$$\mathbf{E} = \begin{bmatrix} (\mathbf{C}\bar{\mathbf{R}}^{-1})^T \\ \mathbf{d}^T \end{bmatrix}, \mathbf{f} = [0 \quad \dots \quad 0 \quad 1]^T \quad (10a)$$

$$\min_{\mathbf{u}} \|\mathbf{E}\mathbf{u} - \mathbf{f}\| \quad \text{s.t.} \quad \mathbf{u} \geq \mathbf{0} \quad (10b)$$

$$\mathbf{y} = -\frac{\boldsymbol{\varepsilon}(1 : \text{col}_E - 1)}{\boldsymbol{\varepsilon}(\text{col}_E)} \quad (10c)$$

$$\mathbf{x} = \bar{\mathbf{R}}^{-1}\mathbf{y} \quad (10d)$$

$$\boldsymbol{\varepsilon} = \mathbf{f} - \mathbf{E}\mathbf{u} \quad (10e)$$

This approach with passivity assessment and subsequent passivity enforcement is repeated until all passivity violations have been removed. An overview of the computational flow of *RPdriver* is shown in Fig. 2.

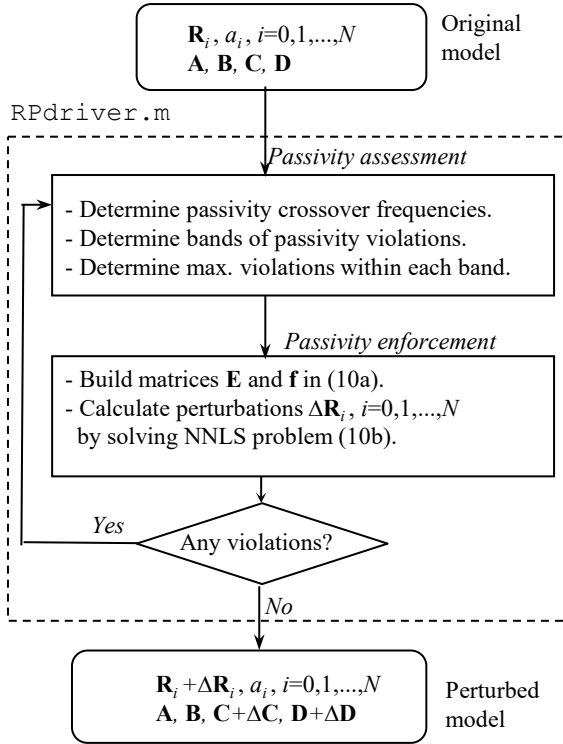


Fig. 2. Enforcing passivity using *RPdriver*.

B. User Interaction

The user can control the behavior of *RPdriver* by a set of control parameters,

- Least squares weighting options as in *VFdriver*, see Table I.
- Option for establishing crossover frequencies using array of user-provided frequencies, $[\omega_1 \ \omega_2 \ \dots \ \omega_\Omega]$ instead of test matrix.
- Option for restarting *RPdriver* with a change in passivity assessment method from sweeping to test matrix (or vice versa), without need for recalculating \mathbf{A} in (8a) and the following (blockwise) QR factorization.
- Max. number of iterations.
- Optional use of an inner iteration loop for adding extra constraints where new violations appear.
- Subset of residue matrices to be perturbed.
- Subset of residue matrix elements to be perturbed.
- Scaling factor α and offset β for the passivity constraint,

$$\mathbf{Cx} > \alpha \mathbf{d} + \beta \quad (11)$$

- Complex or real-valued output matrices.
- Choice of NNLS solver.
lsqnonneg: Matlab native solver.
tntnn [33]: downloadable from github.
nnls [31]: included.

The purpose of the control parameters α and β in (11) is to reduce the number of iterations needed for removing passivity violations. By specifying $\alpha > 1$ and $\beta > 0$, larger corrections

are made. The use of β is demonstrated in Section X.

Regarding the three alternative LS solvers, not much difference is observed for small-scale and medium-size problems. For large-scale problems, *tntnn* and in particular *nnls* can be much faster than *lsqnonneg* if there are many passivity violations [20].

V. CONTROL SETTINGS AND CPU TIME

For cases with high orders and many terminals/ports, the calculation time may become substantial. The computation time is also dependent on the applied control settings as explained below.

A. VFdriver

The initial pole relocations by fitting diagonal elements \mathbf{h}_1 is much faster than the fitting of the full matrix elements \mathbf{h}_2 . Therefore, the number of iterations for fitting \mathbf{h}_2 should for large problems be restricted, or even skipped, e.g. by using a slightly higher model order than strictly necessary. Following the pole relocations (by fitting \mathbf{h}_1 and \mathbf{h}_2), a final calculation of the residue matrices $\{\mathbf{R}_i\}$ is performed. The computation time for the residue matrices is in particular short when unitary weighting is chosen, because the residue matrices are then solved as a linear system with multiple right sides. A further improvement in efficiency is achieved by specifying the residue calculation to be performed using Normal Equations (instead of default QR decomposition).

B. RPdriver

The CPU time of *RPdriver* is usually dominated by the passivity assessment step within the iteration loop. The number of loop counts can be reduced by increasing the correction step length, via parameters α and β in (11).

The default method of passivity assessment is based on the eigenvalues of a test matrix which is large in cases with many terminals/ports and/or high orders. The cubic time for eigenvalue computation makes this calculation slow for large cases. The CPU time problem is avoided by instead basing the passivity assessment on a set of user-provided frequency samples. If desired, *RPdriver* can afterwards be called a second time with use of the passivity test matrix (and few iterations), to ensure that any (minor) passivity violations are removed.

The passivity enforcement step is always fast when unitary weighting is specified. With a weighting scheme which gives different weighting of the individual matrix elements (e.g., inverse magnitude weighting), the QR decomposition of the block-diagonal \mathbf{A} in (8a) becomes slow for problems with many terminals because the blocks become different, requiring all blocks to be subjected to QR factorization. This factorization step is performed only a single time, when starting *RPdriver*. In the case that *RPdriver* is to be started again (e.g. changing passivity assessment from frequency samples to test matrix), the user has an option to specify continued calculation, by which the QR-decomposition from the previous call to *RPdriver* is re-used.

VI. TEST CASES

To demonstrate the application of *VFdriver* and *RPdriver* and some of their control settings, four test cases are presented which are taken from electric high-voltage power systems and high-speed electronics modeling. The test cases can be downloaded from [26].

Table II shows the main characteristics of the four data sets in terms of number of terminals/ports P , number of frequency samples K , parameter type (Y , Z , or S), as well as the model order N that will be used in the modeling.

TABLE II
TEST CASES.

Case	P	K	N	Parameter
1. Transformer black-box	6	338	120	Y
2. FDNE	33	2501	200	Y
3. Transformer white-box	159	14	8	Z
4. High-speed interconnect	120	900	24, 40	S

1. The transformer black-box data set is a 6×6 terminal admittance matrix $\mathbf{Y}(\omega)$, obtained by frequency sweep measurements on a three-winding three-phase power transformer. Information about the transformer and the measurements are given in CIGRE TB904 [35].
2. The FDNE data set is a 33×33 terminal admittance matrix $\mathbf{Y}(\omega)$ which characterizes the terminal behavior of a high-voltage power system in the French grid. The matrix has been obtained via frequency domain calculations in EMTP. (Contribution from Elias Mberou, RTE, France).
3. The transformer white-box data set is a 159×159 impedance matrix $\mathbf{Z}(\omega)$, being the branch impedance matrix of a 3-winding 1-ph transformer, calculated using FEM. (Contribution from Luiz Fernando de Oliveira, Brazil).
4. The high-speed interconnect data set is a 120×120 scattering matrix $\mathbf{S}(\omega)$ which characterizes the port behavior of a chip-to-chip interconnect. The matrix has been obtained via calculations using Keysight Momentum. (Contribution from Prof. Chiu-Chih Chou, Taiwan ROC).

All calculations are performed with Matlab R2023b on a 64-bit Windows 11 laptop with 13th Gen Intel(R) Core(TM) i7-1370P at 1.90 GHz, and 32 GB RAM.

VII. TRANSFORMER BLACK-BOX MODEL (Y-PARAMETERS)

This first example demonstrates how the computational effort and convergence properties of *RPdriver* is affected by the choice of LS weighting scheme and the optional usage of an inner loop in the passivity assessment/enforcement iteration. The example has $P=6$ terminals, given at $K=338$ logarithmically spaced frequencies between 50 Hz and 10 MHz.

A. Vector Fitting

Using *VFdriver*, a model with $N = 120$ pole-residue terms is fitted to the data using $M_1 = 7$ initial iterations and $M_2 = 4$ final iterations. The calculations use inverse magnitude

weighting and enforcement of asymptotic passivity. Fig. 3 shows the magnitude function of the original data set (36 traces) and of the fitted model. The calculations were performed in 0.83 sec. For comparison, with unitary weighting the CPU time is similar: 0.80 sec.

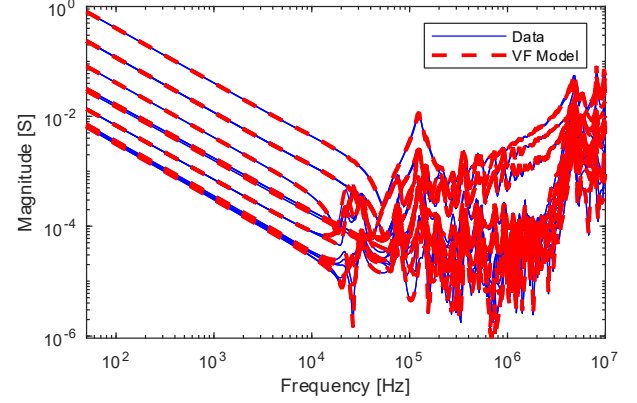


Fig. 3. Admittance matrix elements, fitted using *VFdriver*.

B. Passivity Enforcement

The model obtained by *VFdriver* may have passivity violations. In order to check for passivity violations and remove them, a call to *RPdriver* is made. The call parameters specify passivity assessment by use of test matrix, and passivity enforcement with use of inverse magnitude weighting and no inner loop iterations. *RPdriver* identified several frequency bands with passivity violations, and the violations were removed from the model. Fig. 4 shows a plot of the eigenvalues of $\mathbf{G}(\omega)$ generated by *RPdriver*. It is observed that the presence of negative eigenvalues have been removed, thereby verifying that the perturbed model is positive.

The contributors to the CPU time consumption are listed in Table III. The table also shows the CPU time with use of unitary weighting. In both cases, passivity assessment is the biggest contributor to CPU time. It is further observed that use of inverse magnitude weighting requires more iterations to remove the passivity violations than unitary weighting. The reason is that with inverse magnitude weighting, each residue perturbation step tends to cause new violations to arise.

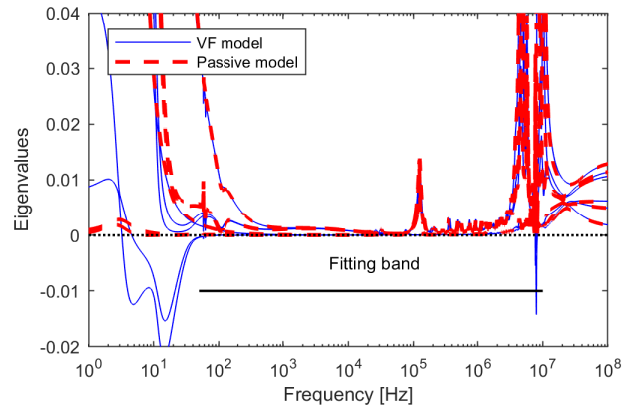


Fig. 4. Passivity enforcement using *RPdriver*. Eigenvalues of $\mathbf{G}(\omega)$.

TABLE III
PASSIVITY ENFORCEMENT ($P=12, K=751, N=150$).

	Inverse magnitude	Unitary
Number of iterations	16	3
Passivity assessment	2.46 sec	0.719 sec
Passivity enforcement	0.199 sec	0.084 sec

The robustness of the iterations can be improved by making use of the inner loop feature where additional constraints are added at those frequencies where new violations will appear. Fig. 5 shows the number of outer loop iterations, without inner loop iterations and with use of two inner loop iterations. As expected, the inner loop iterations result in a more consistent decay of the maximum passivity violation. But each inner loop iteration requires to assess passivity violations one additional time, thereby increasing the total CPU time. With inverse magnitude weighting and two inner loop iterations, the total CPU time increased to 5.2 sec, compared to 2.7 sec without inner loop iterations.

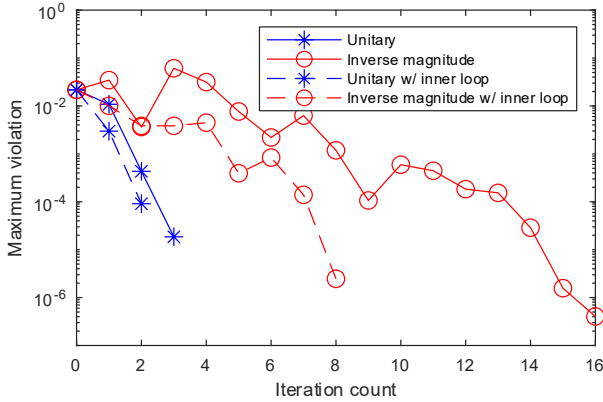


Fig. 5. Passivity enforcement using *RPdriver*. Maximum passivity violation vs. outer loop iteration.

VIII. FDNE MODEL (Y-PARAMETERS)

This second example demonstrates how the CPU time by *VFdriver* can be reduced by reducing the number of M_2 pole relocations, and how the CPU time by *RPdriver* can be reduced by specifying passivity assessment based on frequency samples. The example has eleven 3-ph buses, i.e., $P=33$, given at $K=2501$ linearly spaced frequencies between 0.1 Hz and 2500 Hz. The admittance data is to be fitted using $N=200$ pole-residue terms with inverse magnitude weighting for both the Vector Fitting and passivity enforcement.

A. Vector Fitting

One particular issue with the data case is that several elements are zero, as observed in the sparsity plot in Fig. 6. The zero elements lead to infinite LS weights with use of inverse magnitude weighting, causing failure of the calculation. This issue can be remedied by use of the user-provided weighting array, by setting a finite number for the weight of zero elements. In this example we use a different approach whereby we simply replace the zero elements by a small non-zero number, in this

case $y(\omega)=10^{-12}$. Fig. 7 shows the rational fitting result with use of *VFdriver* with $M_1=10$ initial iterations and $M_2=4$ final iterations. The calculations required a total of 182 sec.

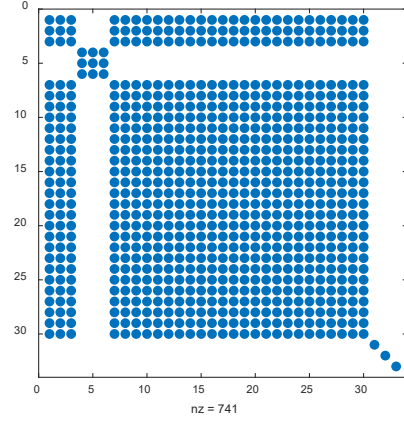


Fig. 6. Sparsity plot of $Y(\omega)$.

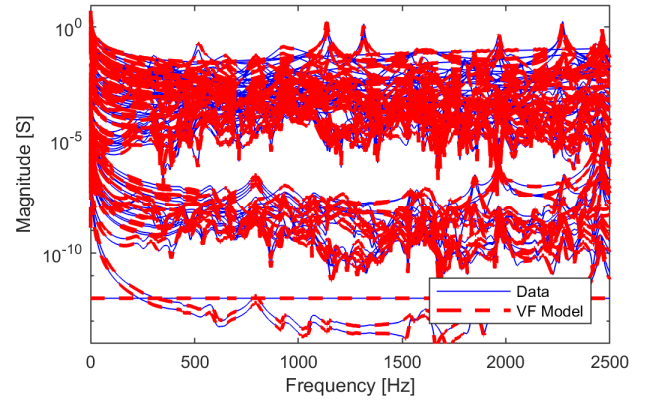


Fig. 7. Admittance matrix elements, fitted using *VFdriver*.

The calculation time is reduced when reducing the number of M_2 final iterations. Table IV shows how the total CPU time and the (weighted) rms error changes as M_2 is reduced from 4 to 0. A substantial decrease in calculation time takes place, at a moderate increase in the fitting error. The reduction in CPU time is in this case insignificant when moving from $M_2=1$ to $M_2=0$ because with $M_2=0$, \mathbf{R}_0 gets negative eigenvalues, thereby requiring an extra residue calculation after enforcement of asymptotic passivity for \mathbf{R}_0 .

TABLE IV
VECTOR FITTING WITH $M_1=10$ ($P=33, K=2501, N=200$).

	$M_2=4$	$M_2=3$	$M_2=2$	$M_2=1$	$M_2=0$
CPU time [sec]	182.2	154.6	89.0	71.5	72.7
Rmserr, weighted	0.027	0.029	0.030	0.033	0.065

B. Passivity Enforcement

The passivity test matrix is in this case very big, with dimension $PN \times PN = 6600 \times 6600$. To avoid excessive

calculation times for the eigenvalues calculation, we base the passivity assessment on frequency samples only. We use samples from DC to 3112 Hz, with the same frequency resolution as the original data set, giving a total of 3125 samples. Passivity was enforced requiring 31 iterations with a total of 113 seconds CPU time, with 84 sec for passivity assessment and 29 sec for the passivity enforcement. For comparison, with use of the test matrix, the calculation time increased from 113 sec to 2339 sec, requiring 34 iterations. With use of the Matlab solver *lsqnonneg* instead of the default *nls* solver, the calculation time was 164 sec with 31 iterations.

Fig. 8 shows how the maximum passivity violation decreases with iteration count, demonstrating a consistent decrease with iteration count, although with superimposed oscillations.

Fig 9 shows a plot of the eigenvalues of $\mathbf{G}(\omega)$, before and after passivity enforcement. The presence of negative eigenvalues has been successfully removed.

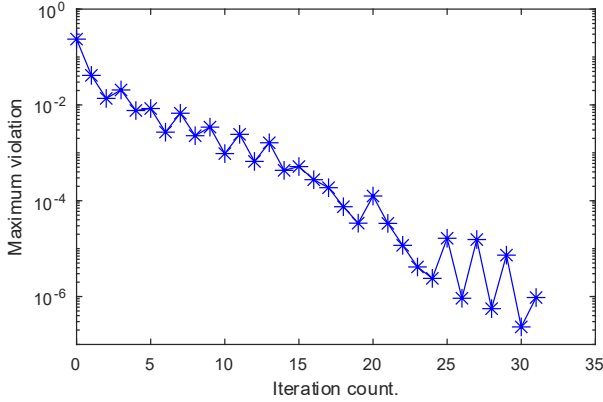


Fig. 8. Maximum passivity violation vs. iteration count.

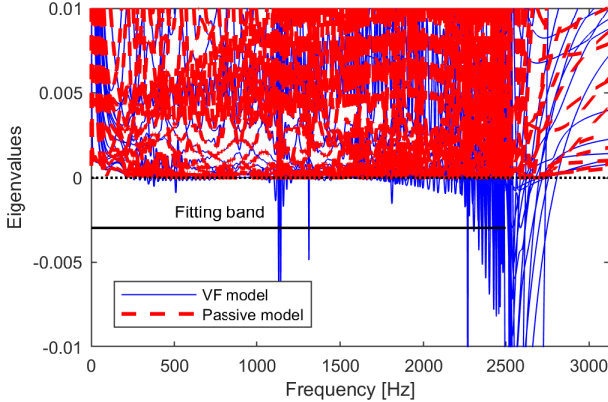


Fig. 9. Passivity enforcement using *RPdriver*. Eigenvalues of $\mathbf{G}(\omega)$.

IX. TRANSFORMER WHITE-BOX MODEL (Z-PARAMETERS)

This third example demonstrates the applicability of the software for passive fitting of smooth responses. In white-box transformer modeling, the frequency dependency of the transformer's $P \times P$ branch impedance matrix $\mathbf{Z}_B(\omega)$ should be taken into account for best possible accuracy [30]. A data set calculated by FEM is considered, with $P=159$ and $K=14$ logarithmically spaced samples between 60 Hz and 1.08 MHz.

Using *VFdriver*, a rational model is calculated using $N=8$ poles with inverse magnitude weighting. The computations are

very fast, thereby allowing many pole relocating iterations to be used. With $M_1=10$ and $M_2=10$, the model extraction was performed in 5.4 sec. The fitting result is shown in Fig. 10.

One passivity violation results at low frequencies. Using *RPdriver* with inverse magnitude weighting, the violation is removed in a single iteration. The calculation time was 2.0 sec for passivity assessment (with use of test matrix), and 0.5 sec for passivity enforcement. Fig. 11 confirms, using a high-resolution plot, that the passivity enforcement has made all eigenvalues of $\mathbf{R} = \text{Re}\{\mathbf{Z}\}$ positive. (With use of a higher fitting order, the initial model by *VFdriver* resulted passive).

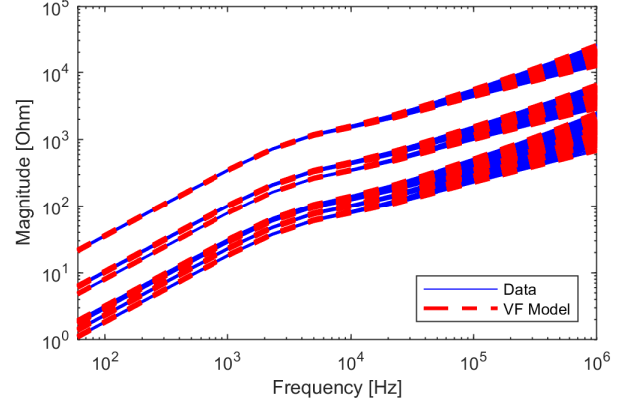


Fig. 10. Impedance matrix elements, fitted using *VFdriver*.

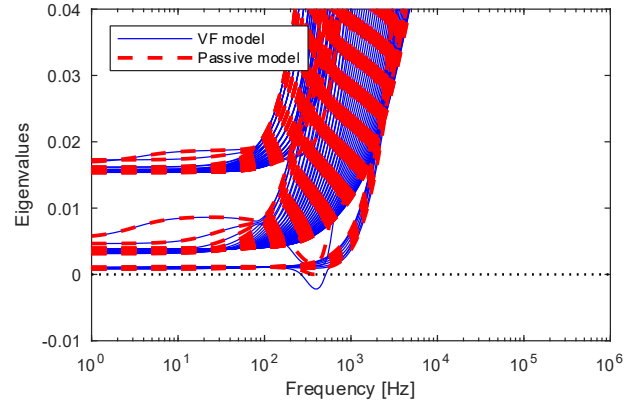


Fig. 11. Passivity enforcement using *RPdriver*. Eigenvalues of $\mathbf{R}(\omega)$.

X. CHIP-TO-CHIP INTERCONNECT MODEL (S-PARAMETERS)

A. Vector Fitting

The last example is an S-parameter data set representing a chip-chip interconnect. The data set $\mathbf{S}(\omega)$ has $P=120$ ports and $K=900$ frequency samples, linearly spaced between 20 MHz and 20 GHz. A pole-residue model is to be calculated using unitary weighting, in both VF and passivity enforcement. Due to the large problem size, we will compare two alternative ways of calculating the model, emphasizing possible savings to the computational cost.

We first fit an initial model to the data using *VFdriver*, with $N=24$ poles, see Table V. In the first alternative, $M_1=7$ pole relocating iterations are applied to the diagonal matrix elements held in \mathbf{h}_1 , followed by $M_2=2$ pole relocating iterations of the

full matrix held in \mathbf{h}_2 . The total calculation time amounts to 19.75 sec, being dominated by the two pole relocations of the full matrix.

In the second alternative, we attempt to reduce computation time by calculating the poles by applying $M_1 = 7$ pole relocating iterations to the diagonal elements only, followed by residue matrix calculation for the full matrix. This approach reduces the calculation time to 2.59 sec (down from 19.75 sec). A side effect is an increase in the rms-error, from 0.00015 to 0.00033.

TABLE V
VF AND PASSIVITY ENFORCEMENT ($P=120, K=900, N=24$).

		Alt. 1	Alt. 2
Approach	Fitting \mathbf{h}_1	7 iterations	7 iterations
	Fitting \mathbf{h}_2	2 iterations	0 iterations
Result	CPU time	19.75 sec	2.59 sec
	rmserr	1.49e-03	3.34e-03

B. Passivity Enforcement

We next proceed with the passivity enforcement, see Table VI. In the first alternative, the passivity assessment is performed using the test matrix. A total of five iterations is needed, requiring 27.68 sec for the passivity assessment and 1.32 sec for the passivity enforcement. In order to reduce computation time, we will for the second alternative perform the passivity assessment using frequency sweeping over 150 samples for determining crossover frequencies, being a mix logarithmically and linearly spaced samples from 0.0001 Hz to 40 GHz, properly covering the fitting band. To reduce the number of required iterations, we also increase the offset parameter β in (11) from the default value of $\beta = 10^{-6}$ to $\beta = 10^{-4}$. This modification increases the step length of the correction. The result is that passivity enforcement is achieved in only two iterations. Finally, a last call to *RPdriver* is used with usage of test matrix for precise passivity assessment, thereby ensuring that no passivity violations are missed out. No further violations were detected. In total, the passivity assessment required 5.69 sec and the passivity enforcement 2.58 sec. The rms error for the two alternative passivity enforcement schemes are seen to be much smaller than the error by the model extraction using *VFdriver* in Table V. Fig. 12 shows the effect of passivity enforcement by Alternative 2 on the singular values σ_j of \mathbf{S} , verifying that all σ_j become smaller than unity.

TABLE VI
COMPUTATION TIME USING *RPDRIVER* ($P=120, K=900, N=24$).

		Alt. 1	Alt. 2
Approach	Passivity assess.	Test matrix	Sweeping + test matrix
	Passivity enf.	$\beta = 10^{-6}$	$\beta = 10^{-4}$
Result	Passivity assess	27.68 sec	5.69 sec
	Passivity enf.	1.32 sec	2.58 sec
	rmserr	4.61e-04	2.14e-4
	N.o. iterations	5	2+0
	Total	48.75 sec	10.86 sec

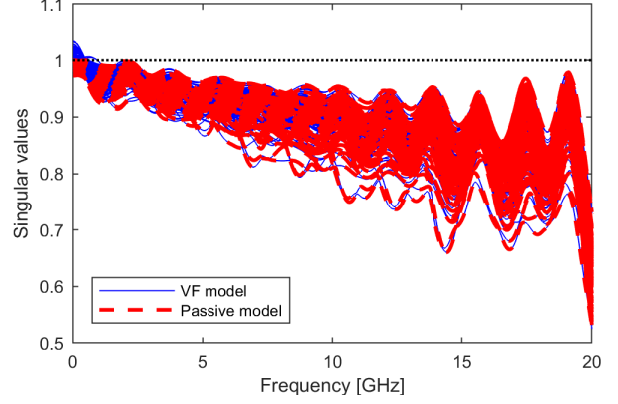


Fig. 12. Passivity enforcement using *RPdriver*. Singular values of $\mathbf{S}(\omega)$.

XI. EXPORT TO TIME DOMAIN CIRCUIT SIMULATORS

Similarly, as with the previous version of MFT, the Y-parameter models can be directly exported to some circuit simulation programs, namely EMTP, PSCAD, and ATP as shown in Table VII. In the new MFT implementation, also Z- and S-parameter models can be exported to PSCAD.

TABLE VII
EXPORT TO CIRCUIT SIMULATORS (Y-PARAMETER MODEL)

Circuit simulator	Parameter type	Export function	Model type
EMTP	Y	<i>netgen_EMTP</i>	Rational model
PSCAD	Y, Z, S	<i>netgen_PSCAD</i>	Rational model
ATP	Y	<i>netgen_ATP</i>	Equivalent circuit

A. Y-Parameters - Transformer Measurements

As an example, we consider the transformer black-box case in Section VII. The final model is exported to the EMTP and PSCAD simulation softwares by calling routines *netgen_EMTP* and *netgen_PSCAD*, respectively. Fig. 13 shows a simulation example where a 1.2/50 μs lightning impulse voltage is applied to one high-voltage (HV) terminal (#1) with the two other HV terminals grounded. The simulation result in Fig. 14 shows the resulting overvoltages on the corresponding low-voltage terminal (#4). The simulation result is shown using EMTP and PSCAD, and compared against an actual measurement, performed by CIGRE JWG A2/C4.52 on this transformer.

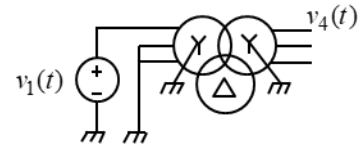


Fig. 13. Simulation of transferred overvoltage on low-voltage terminals.

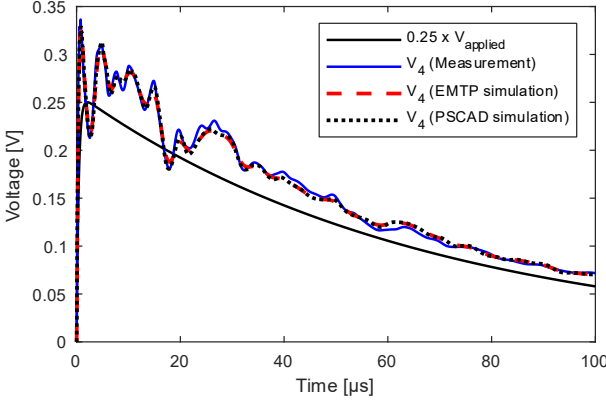


Fig. 14. Simulation of transferred overvoltage on low-voltage terminals.

B. S-parameters – Chip-to-Chip Interconnect

The S-parameter interconnect model for the example in Section X was exported to PSCAD by use of *netgen_PSCAD*. Limitations exist in PSCAD regarding the permitted model size. Therefore, the number of ports in the original data set was limited to $P = 48$ by deleting rows and columns, and a 40th order passive model was fitted to the resulting data set $\mathbf{S}(\omega)$, and exported to PSCAD.

Fig. 15 shows the result from a PSCAD simulation using the FDNE component for inclusion of the model. A 1-Volt step voltage behind 0.001Ω is applied to port 1 with the remaining ports open. The simulation result is shown in Fig. 16, demonstrating the expected voltage doubling in the response on the ditto far end port. The plot also shows the simulation result in Matlab using the S-parameter time domain implementation in [36]. A near identical result is obtained.

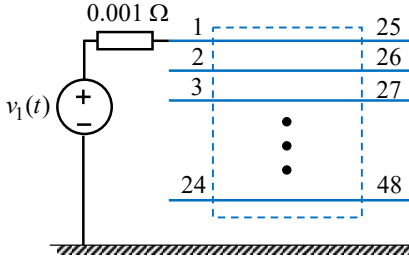


Fig. 15. Simulation of transient voltage response on ports 25-28.

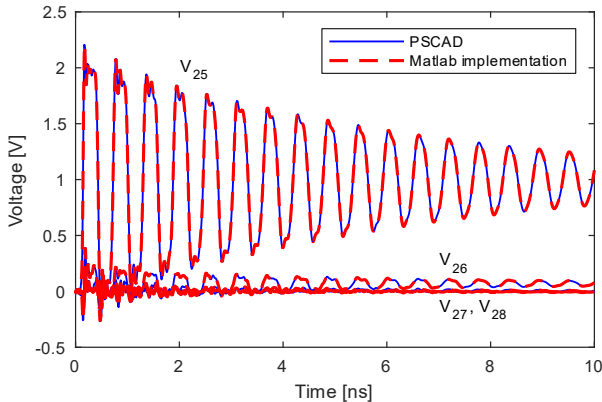


Fig. 16. Simulation of transient voltage response on ports 25-28.

XII. DISCUSSION

A. Vector Fitting

Vector Fitting (VF) iteratively relocates an initial pole set to better positions. To obtain fast computations, one should avoid unnecessary iterations. Routine *VFdriver* fits a rational model to the $P \times P$ symmetrical matrix (\mathbf{Y} , \mathbf{Z} , or \mathbf{S}) in two stages. An improved pole set is first calculated by stacking the P diagonal elements into a vector $\mathbf{h}_1(s)$, which is iteratively fitted by VF with user-defined initial poles. Afterwards, the $P(P+1)/2$ elements of the upper matrix triangle are stacked into a vector $\mathbf{h}_2(s)$ which is fitted using VF with the poles from $\mathbf{h}_1(s)$ as initial poles. The reason for this two-stage separation of the fitting process is that the fitting of the short vector $\mathbf{h}_1(s)$ is much faster than that of $\mathbf{h}_2(s)$, thereby reducing the required number of iterations for $\mathbf{h}_2(s)$.

The use of diagonal elements in $\mathbf{h}_1(s)$ is justified as follows. In order to obtain a good pole set for the final fitting of $\mathbf{h}_2(s)$, the poles in $\mathbf{h}_2(s)$ should be observable in $\mathbf{h}_1(s)$. It can be shown that the sum of the diagonal elements equals the sum of the matrix eigenvalues. This sum (matrix trace) can therefore be used for representing $\mathbf{h}_1(s)$, as it was done in the original MFT. But some fitting problems involve a mix of large and small elements on the matrix diagonal, causing the small elements to be lost in the sum. Usage of all diagonal elements as separate elements in $\mathbf{h}_1(s)$ alleviates this problem at a small additional computational cost, relative to that of fitting $\mathbf{h}_2(s)$.

B. Passivity Enforcement

Routine *RPdriver* works by perturbing the model's residue matrices such that all passivity violations become removed. In some situations, the iterations may fail to converge. Non-convergence can be a result of the non-linear relation between matrix perturbation and the change to the matrix eigenvalues (6a) or singular values (6b). Convergence failure can also be a result of the passivity enforcement causing new passivity violations to arise.

Although the approach attempts to minimize the LS change to the original model, it is possible that the change to the model's behavior is not acceptable. This is in particular an issue in the case of large in-band violations and with usage on inverse magnitude weighting. The user should therefore assess whether the model behavior is acceptable, for instance by comparing simulation results using the original model and the perturbed model.

XIII. CONCLUSION

A new Matrix Fitting Toolbox (MFT-NNLS) has been presented, for passive macromodeling from frequency responses that can represent admittance, impedance, or scattering data. The applicability to large-scale cases and the rich set of control parameters makes it suitable for many topical areas within power systems and high-speed electronics. The model export capability for EMTP, PSCAD and ATP should

make the extracted models particularly attractive for researchers and practicing engineers in the power systems area.

XIV. ACKNOWLEDGEMENT

The author would like to thank the members of CIGRE JWG A2/C4.52 for providing the transformer measurement used in Section VII, Elias Mberou (RTE, France) for providing the FDNE example in Section VIII, Luiz Fernando de Oliveira (Hitachi, Brazil) for providing the transformer branch impedance example in Section IX, and Prof. Dr. Chiu-Chih Chou (National Central University, Zhong-Li, Taiwan, ROC) for providing the interconnect example in Section X.

XV. REFERENCES

- [1] S. Grivet-Talocia and B. Gustavsen, *Passive Macromodeling: Theory and Applications*, John Wiley and Sons, 2015.
- [2] C.K. Sanathanan, and J. Koerner, "Transfer function synthesis as a ratio of two complex polynomials", *IEEE Trans. Automatic Control*, vol. 8, no. 1, pp. 56-58, Jan. 1963.
- [3] B. Gustavsen, and A. Semlyen, "Rational approximation of frequency domain responses by vector fitting", *IEEE Trans. Power Delivery*, vol. 14, no. 3, pp. 1052-1061, July 1999.
- [4] B. Gustavsen, "Improving the pole relocating properties of vector fitting", *IEEE Trans. Power Delivery*, vol. 21, no. 3, pp. 1587-1592, July 2006.
- [5] D. Deschrijver, M. Mrozowski, T. Dhaene, and D. De Zutter, "Macromodeling of multiport systems using a fast implementation of the vector fitting method", *IEEE Microwave and Wireless Components Letters*, vol. 18, no. 6, pp. 383-385, June 2008.
- [6] A. Chinae and S. Grivet-Talocia, "On the parallelization of Vector Fitting algorithms," in *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 1, no. 11, pp. 1761-1773, Nov. 2011.
- [7] S. Ganeshan, N. K. Elumalai, R. Achar and W. K. Lee, "GVF: GPU-based Vector Fitting for modeling of multiport tabulated data networks," in *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 10, no. 8, pp. 1375-1387, Aug. 2020.
- [8] V. Kukutla, R. Achar and W. K. Lee, "TC-GVF: Tensor core GPU-based Vector Fitting via accelerated tall-skinny QR Solvers," in *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 15, no. 1, pp. 54-63, Jan. 2025.
- [9] B. Gustavsen and C. Heitz, "Fast realization of the modal vector fitting method for rational modeling with accurate representation of small eigenvalues", *IEEE Trans. Power Delivery*, vol. 24, no. 3, pp. 1396-1405, July 2009.
- [10] A. Carlucci, T. Bradde and S. Grivet-Talocia, "Improving accuracy of rational macromodels under realistic loading conditions," *2023 IEEE 32nd Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS)*, Milpitas, CA, USA, 2023, pp. 1-3.
- [11] A. Ramirez, B. Gustavsen and I. Ramirez, "Rational Approximation of Nonlinear Systems Via NL-VF," in *IEEE Transactions on Power Delivery*, vol. 38, no. 3, pp. 1918-1926, June 2023.
- [12] S. Lefteriut and A. C. Antoulas, "A new approach to modeling multiport systems from frequency-domain data", *IEEE Trans Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 1, pp. 14-27, Jan. 2010.
- [13] M. Berljafa and S. Güttel, "The RKFIT algorithm for nonlinear rational approximation", *SIAM J. Sci. Comput.* **39**(5), A2049–A2071 (2017).
- [14] Y. Nakatsukasa, O. Sète, and L.N. Trefethen, "The AAA algorithm for rational approximation", *SIAM J. Sci. Comput.* **40**(3), A1494–A1522, 2018.
- [15] B. Gustavsen, and A. Semlyen, "Enforcing passivity for admittance matrices approximated by rational functions", *IEEE Trans. Power Systems*, vol. 16, no. 1, pp. 97-104, Feb. 2001.
- [16] D. Saraswat, R. Achar, and M. S. Nakhla, "Global passivity enforcement algorithm for macromodels of interconnect subnetworks characterized by tabulated data," *IEEE Trans. Very Large Scale (VLSI) Syst.*, vol. 13, no. 7, pp. 819–832, July 2005.
- [17] B. Gustavsen, "Fast passivity enforcement for pole-residue models by perturbation of residue matrix eigenvalues", *IEEE Trans. Power Delivery*, vol. 23, no. 4, pp. 2278-2285, October 2008.
- [18] S. Grivet-Talocia, "A perturbation scheme for passivity verification and enforcement of parameterized macromodels," *IEEE Trans. Components, Packaging and Manufacturing Technology*, vol. 7, no. 11, pp. 1869-1881, Nov. 2017.
- [19] B. Gustavsen, "Passivity enforcement by residue perturbation via constrained non-negative least squares", *IEEE Trans. Power Delivery*, vol. 36, no. 5, pp. 2758-2767, October 2021.
- [20] B. Gustavsen, "An efficient residue perturbation scheme for passivity enforcement of S-parameter rational models", *IEEE Trans. Electromagnetic Compatibility*, vol. 67, no. 3, pp. 913-920, June 2025.
- [21] B. Gustavsen, "Passivity enforcement of rational models via modal perturbation", *IEEE Trans. Power Delivery*, vol. 23, no. 2, pp. 768-775, April 2008.
- [22] A. Carlucci, T. Bradde and S. Grivet-Talocia, "Improving robustness to termination conditions in passivity enforcement of rational macromodels," in *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. 15, no. 1, pp. 64-74, Jan. 2025.
- [23] S. Grivet-Talocia, "An adaptive sampling technique for passivity characterization and enforcement of large interconnect macromodels," *IEEE Transactions on Advanced Packaging*, vol. 30, no. 2, pp. 226-237, May 2007.
- [24] A. Semlyen and B. Gustavsen, "A half-size singularity test matrix for fast and reliable passivity assessment of rational models", *IEEE Trans. Power Delivery*, vol. 24, no. 1, pp. 345-351, Jan. 2009.
- [25] B. Gustavsen and A. Semlyen, "Fast passivity assessment for S-parameter rational models via a half-size test matrix", *IEEE Trans. Microwave Theory And Techniques*, vol. 56, no. 12, pp. 2701-2708, December 2008.
- [26] The vector fitting web site, <https://www.sintef.no/projectweb/vectorfitting/>
- [27] B. Gustavsen, "Computer code for rational approximation of frequency dependent admittance matrices", *IEEE Trans. Power Delivery*, vol. 17, no. 4, pp. 1093-1098, October 2002.
- [28] B. Gustavsen "Computer code for passivity enforcement of rational macromodels", *IEEE Trans. Advanced Packaging*, vol. 30, no. 2, pp. 209-215, May 2007.
- [29] U. D. Annakkage et al., "Dynamic system equivalents: A survey of available techniques," *IEEE Trans. Power Delivery*, vol. 27, no. 1, pp. 411-420, Jan. 2012.
- [30] CIGRE TB 900, "High-frequency transformer and reactor models for network studies. Part A – White box models", CIGRE JWG A2/C4.52, April 2023.
- [31] Bill Whiten (2025), "Nnls-Non negative least squares", (<https://www.mathworks.com/matlabcentral/fileexchange/38003-nnls-non-negative-least-squares>), MATLAB Central File Exchange. Retrieved March 17, 2025.
- [32] C.L. Lawson and R.J. Hanson, *Solving least squares problems*, SIAM, 1995. ISBN 0-89871-356-0.
- [33] J.M. Myhre, E. Frahm, D.J. Lilja, and Mo.O. Saar, "TNT-NN: A fast active set method for solving large non-negative least squares problems", *Procedia Computer Science*, vol. 108, pp. 755-764, 2017.
- [34] G.H. Golub and C.F. Van Loan, *Matrix Computations*, John Hopkins University Press, Second Ed., 1989. ISBN 0-8018-3772-3.
- [35] CIGRE TB 904, "High-frequency transformer and reactor models for network studies. Part E – Measurements and transformer design details", CIGRE JWG A2/C4.52, April 2023.
- [36] B. Gustavsen and H.M.J. De Silva, "Inclusion of rational models in an electromagnetic transients program – Y-parameters, Z-parameters, S-parameters, transfer functions", *IEEE Trans. Power Delivery*, vol. 28, no. 2, pp. 1164-1174, April 2013.

BIOGRAPHY

Bjørn Gustavsen (M'94–SM'2003–F'2014) was born in Norway in 1965. He received the M.Sc. degree and the Dr. Ing. degree in Electrical Engineering from the Norwegian Institute of Technology in Trondheim, Norway, in 1989 and 1993, respectively. Since 1994 he has been working at SINTEF Energy Research, currently as Chief Research Scientist. He is also an adjunct Professor at NTNU, since 2020. His interests include simulation of electromagnetic transients and modeling of frequency dependent effects. He spent 1996 as a Visiting Researcher at the University of Toronto, Canada, and the summer of 1998 at the Manitoba HVDC Research Centre, Winnipeg, Canada. He was Marie Curie Fellow at the University of Stuttgart, Germany, August 2001–August 2002.