Modal Approximation and Computation of Dominant Poles

Joost Rommes

NXP Semiconductors/Corp. I&T/DTF
High Tech Campus 37
5656 AE Eindhoven
The Netherlands
joost.rommes@nxp.com

1 Introduction

A large scale dynamical system can have a large number of modes. Like a general square matrix can be approximated by its largest eigenvalues, i.e. by projecting it onto the space spanned by the eigenvalues corresponding to the largest eigenvalues, a dynamical system can be approximated by its dominant modes: a reduced order model, called the *modal equivalent*, can be obtained by projecting the state space on the subspace spanned by the dominant modes. This technique, *modal approximation* or *modal model reduction*, has been successfully applied to transfer functions of large-scale power systems, with applications such as stability analysis and controller design, see [16] and references therein.

The dominant modes, and the corresponding dominant poles of the system transfer function, are specific eigenvectors and eigenvalues of the state matrix. Because the systems are very large in practice, it is not feasible to compute all modes and to select the dominant ones. This chapter is concerned with the efficient computation of these dominant poles and modes specifically, and their use in reduced order modeling. In Sect. 2 the concept of dominant poles and modal approximation is explained in more detail. Dominant poles can be computed with specialized eigensolution methods, as is described in Sect. 3. Some generalizations of the presented algorithms are shown in Sect. 4. The theory is illustrated with numerical examples in Sect. 5 and 6 concludes.

Part of the contents of this chapter is based on [15, 16]. The pseudocode algorithms presented in this chapter are written using Matlab-like [21] notation.

2 Transfer Functions, Dominant Poles and Modal Equivalents

Throughout this section and the next, only single-input single-output (SISO) transfer functions are considered. In Sect. 4, the theory is generalized to multi-input multi-output (MIMO) transfer functions.

The transfer function of a SISO linear, time invariant system

$$\begin{cases} \dot{\mathbf{x}}(t) = A\mathbf{x}(t) + \mathbf{b}u(t) \\ y(t) = \mathbf{c}^*\mathbf{x}(t) + du(t), \end{cases}$$

where $A \in \mathbb{R}^{n\times n}$, $\mathbf{x}(t), \mathbf{b}, \mathbf{c} \in \mathbb{R}^n$ and $u(t), y(t), d \in \mathbb{R}$, is defined as

$$H(s) = \mathbf{c}^*(sI - A)^{-1}\mathbf{b} + d, \tag{1}$$

where $I \in \mathbb{R}^{n\times n}$ is the identity matrix and $s \in \mathbb{C}$.

The eigenvalues $\lambda_i \in \mathbb{C}$ of the matrix A are the poles of transfer function (1). An eigentriplet $(\lambda_i, \mathbf{x}_i, \mathbf{y}_i)$ is composed of an eigenvalue λ_i of A and the corresponding right and left eigenvectors $\mathbf{x}_i, \mathbf{y}_i \in \mathbb{C}^n$:

$$\begin{aligned} A\mathbf{x}_i &= \lambda_i\mathbf{x}_i, \qquad \mathbf{x}_i \neq 0, \\ \mathbf{y}_i^*A &= \lambda_i\mathbf{y}_i^*, \qquad \mathbf{y}_i \neq 0. \end{aligned}$$

Assuming that A is a nondefective matrix, the right and left eigenvectors can be scaled so that $\mathbf{y}_i^*\mathbf{x}_i = 1$. Furthermore, it can be shown that left and right eigenvectors corresponding to distinct eigenvalues are orthogonal: $\mathbf{y}_i^*\mathbf{x}_j = 0$ for $i \neq j$. The transfer function $H(s)$ can be expressed as a sum of residues R_i over first order poles [10]:

$$H(s) = \sum_{i=1}^{n} \frac{R_i}{s - \lambda_i} + d, \tag{2}$$

where the residues R_i are

$$R_i = (\mathbf{c}^*\mathbf{x}_i)(\mathbf{y}_i^*\mathbf{b}).$$

A possible definition of a dominant pole follows from inspection of (2):

Definition 1. *A pole λ_i of $H(s)$ with corresponding right and left eigenvectors $\mathbf{x}_i$ and $\mathbf{y}_i$ ($\mathbf{y}_i^*\mathbf{x}_i = 1$) is called dominant if $\widehat{R}_i$=$|R_i|/|Re(\lambda_i)|$=$|(\mathbf{c}^*\mathbf{x}_i)(\mathbf{y}_i^*\mathbf{b})|/|Re(\lambda_i)|$ is relatively large compared to $\widehat{R}_j$, $j \neq i$.*

The quantity $\widehat{R}_i$ will be referred to as the dominance index of pole λ_i. It follows from this definition that a dominant pole is well observable and controllable. This can also be observed from the Bode magnitude plot of $H(s)$, where peaks occur at frequencies close to the imaginary parts of the dominant poles of $H(s)$. If the poles are ordered to decreasing $\widehat{R}_i$, a so called transfer function modal equivalent can be defined as follows.

Definition 2. *A transfer function modal equivalent $H_k(s)$ is an approximation of a transfer function $H(s)$ that consists of $k < n$ terms:*

$$H_k(s) = \sum_{j=1}^{k} \frac{R_j}{s - \lambda_j} + d. \tag{3}$$

A modal equivalent that consists of the most dominant terms determines the effective transfer function behavior [20]. If $X \in \mathbb{C}^{n\times k}$ and $Y \in \mathbb{C}^{n\times k}$ are matrices having the left and right eigenvectors $\mathbf{y}_i$ and $\mathbf{x}_i$ of A as columns, such that $Y^*AX = \Lambda = \text{diag}(\lambda_1, \ldots, \lambda_k)$, with $Y^*X = I$, then the corresponding (complex) reduced system follows by setting $\mathbf{x} = X\widetilde{\mathbf{x}}$:

$$\begin{cases} \dot{\widetilde{\mathbf{x}}}(t) = \Lambda\widetilde{\mathbf{x}}(t) + (Y^*\mathbf{b})u(t) \\ \widetilde{\mathbf{y}}(t) = (\mathbf{c}^*X)\widetilde{\mathbf{x}}(t) + du(t). \end{cases}$$

For stable systems, the error in the modal equivalent can be quantified as [7]

$$\begin{aligned} \|H - H_k\|_\infty &= \| \sum_{j=k+1}^{n} \frac{R_j}{s-\lambda_j} \|_\infty \\ &\leq \sum_{j=k+1}^{n} \frac{|R_j|}{|\text{Re}(\lambda_j)|}, \end{aligned}$$

where $\|H\|_\infty$ is the operator norm induced by the 2-norm in the frequency domain [2,7]. An advantage of modal approximation is that the poles of the modal equivalent are also poles of the original system.

It should be stressed that there are more definitions of a dominant pole, see [1, 7, 22]. The definition often depends on the application: in stability analysis for instance, the poles with positive real parts are considered as the dominant poles. Throughout this chapter, Def. 1 is used.

3 Computing Dominant Poles

3.1 Introduction

The poles of transfer function (1) are the $\lambda \in \mathbb{C}$ for which $\lim_{s\to\lambda} |H(s)| = \infty$. Consider now the function

$$G : \mathbb{C} \longrightarrow \mathbb{C} : s \mapsto \frac{1}{H(s)}. \tag{4}$$

For a pole λ of $H(s)$, $\lim_{s\to\lambda} G(s) = 0$. In other words, the poles are the roots of $G(s)$ and a good candidate to find these roots is Newton's method. This idea is the basis of the Dominant Pole Algorithm (DPA) [11]. Because the direct transmission term d has no influence on the dominance of a pole, $d = 0$ unless stated otherwise.

3.2 Dominant Pole Algorithm (DPA)

The derivative of $G(s)$ (4) with respect to s is given by

$$G'(s) = -\frac{H'(s)}{H^2(s)}. \tag{5}$$

The derivative of $H(s)$ (1) to s is

$$H'(s) = -\mathbf{c}^*(sI - A)^{-2}\mathbf{b}, \tag{6}$$

where it is used that the derivative of the inverse of a square matrix $A(s)$ is given by $d[A^{-1}(s)]/ds = -A^{-1}(s)A'(s)A^{-1}(s)$. Equations (5) and (6) lead to the following Newton scheme:

$$\begin{aligned} s_{k+1} &= s_k - \frac{G(s_k)}{G'(s_k)} \\ &= s_k + \frac{1}{H(s_k)}\frac{H^2(s_k)}{H'(s_k)} \\ &= s_k - \frac{\mathbf{c}^*(s_kI - A)^{-1}\mathbf{b}}{\mathbf{c}^*(s_kI - A)^{-2}\mathbf{b}}. \end{aligned} \tag{7}$$

The formula (7) was originally derived in [3]. Using $\mathbf{x} = (s_kI - A)^{-1}\mathbf{b}$ and $\mathbf{y} = (s_kI - A)^{-*}\mathbf{c}$, an implementation of this Newton scheme is Alg. 1, also known as the Dominant Pole Algorithm (DPA) [11]. The two linear systems that need to be solved in step 3 and 4 of Alg. 1 can be efficiently solved using one LU-factorization $LU = s_kI - A$, by noting that $U^*L^* = (s_kI - A)^*$. It will be assumed in this chapter that an exact LU-factorization is available, although this may not always be the case for real-life examples. If an exact LU-factorization is not available, one has to use inexact Newton schemes, such as Jacobi-Davidson style methods [9, 19].

For nondefective A, Alg. 1 converges asymptotically quadratically, but the solution depends on the initial estimate and hence is not guaranteed to be the most dominant pole. For defective A, the algorithm may fail, because the left and right eigenvector of a defective eigenvalue are orthogonal and hence $\mathbf{y}^*\mathbf{x} \to 0$ in step 5 of Alg. 1. The update in step 5 can be written as the two-sided Rayleigh quotient [14]

$$s_{k+1} = \frac{\mathbf{y}^*A\mathbf{x}}{\mathbf{y}^*\mathbf{x}}. \tag{8}$$

3.3 Subspace Accelerated Dominant Pole Algorithm

While DPA computes a single dominant pole, in practice usually more dominant poles are wanted. The Subspace Accelerated Dominant Pole Algorithm (SADPA) [16] is a generalization of DPA to compute more than one dominant pole. SADPA has three major improvements compared to DPA. Firstly, it uses subspace acceleration, a well-known technique for iterative methods. Secondly, a new selection strategy is used to select the most dominant pole approximation and corresponding right and left eigenvector approximation every iteration. Thirdly, deflation is used to avoid convergence to eigentriplets that are already found. The ideas, leading to SADPA (Alg. 2), are described in the following subsections.

Algorithm 1: The Dominant Pole Algorithm (DPA)

INPUT: System $(A, \mathbf{b}, \mathbf{c})$, initial pole estimate $s_1 \in \mathbb{C}$, tolerance $\varepsilon \ll 1$

OUTPUT: Approximate dominant pole λ and corresponding right and left eigenvectors $\mathbf{x}$ and $\mathbf{y}$

1: Set $k = 1$

2: **while** not converged **do**

3: Solve $\mathbf{x} \in \mathbb{C}^n$ from

$$(s_k I - A)\mathbf{x} = \mathbf{b}$$

4: Solve $\mathbf{y} \in \mathbb{C}^n$ from

$$(s_k I - A)^* \mathbf{y} = \mathbf{c}$$

5: Compute the new pole estimate

$$s_{k+1} = s_k - \frac{\mathbf{c}^* \mathbf{x}}{\mathbf{y}^* \mathbf{x}}$$

6: The pole $\lambda = s_{k+1}$ has converged if

$$\|A\mathbf{x} - s_{k+1}\mathbf{x}\|_2 < \varepsilon$$

7: Set $k = k + 1$

8: **end while**

Subspace Acceleration

A drawback of DPA is that information obtained in the current iteration is discarded at the end of the iteration. The only information that is preserved is contained in the new pole estimate s_{k+1}. The vectors $\mathbf{x}$ and $\mathbf{y}$, however, also contain information about other dominant eigentriplets (i.e., components in the direction of the corresponding eigenvectors) and the idea is to use this information as well. Reasoning this way leads to a generalization of DPA.

A global overview of SADPA is shown in Alg. 2. Starting with an estimate s_1, the first iteration is equivalent to the first iteration of DPA, but instead of discarding the corresponding right and left eigenvector approximations $\mathbf{x}_1$ and $\mathbf{y}_1$, they are kept in spaces X and Y. In the next iteration, these spaces are expanded orthogonally (step 5-6), by modified Gram-Schmidt (MGS) [6], with the approximations $\mathbf{x}_2$ and $\mathbf{y}_2$ corresponding to the new estimate s_2. In Sect. 3.3 it is explained how this new pole estimate is computed. The subspaces grow in dimension and may contain better approximations. This idea is known as subspace acceleration.

In the k-th iteration, k approximations $\widehat{\lambda}_i$ of the dominant poles are found by computing the eigentriplets of the projected matrix pencil (Y^*AX, Y^*X) (step 7-8). The question now is to determine which of these k approximations to use as estimate s_{k+1} in the next iteration.

Algorithm 2: Subspace Accelerated DPA

INPUT: System $(A, \mathbf{b}, \mathbf{c})$, initial pole estimate s_1 and the number of wanted poles p_{max}, tolerance $\varepsilon \ll 1$

OUTPUT: Approximate dominant pole triplets $(\lambda_i, \mathbf{r}_i, \mathbf{l}_i)$, $i = 1, \ldots, p_{max}$

1: $k = 1$, $p_{found} = 0$, $\Lambda = [\,]^{1\times 0}$, $R = L = X = Y[\,]^{n\times 0}$ ($[\,]^{n\times 0}$ denotes an empty matrix of size $n \times 0$)
2: **while** $p_{found} < p_{max}$ **do**
3: Solve $\mathbf{x} \in \mathbb{C}^n$ from

$$(s_k I - A)\mathbf{x} = \mathbf{b}$$

4: Solve $\mathbf{y} \in \mathbb{C}^n$ from

$$(s_k I - A)^* \mathbf{y} = \mathbf{c}$$

5: $X = \text{Expand}(X, R, L, \mathbf{x})$ {Alg. 4}
6: $Y = \text{Expand}(Y, L, R, \mathbf{y})$ {Alg. 4}
7: Compute $T = Y^* A X$ and $G = Y^* X$
8: $(\widehat{\Lambda}, \widehat{X}, \widehat{Y}) = \text{Sort}(T, G, X, Y, \mathbf{b}, \mathbf{c})$ {Alg. 3}
9: **if** $\|A\widehat{\mathbf{x}}_1 - \widehat{\lambda}_1 \widehat{\mathbf{x}}_1\|_2 < \varepsilon$ **then**
10: $(\Lambda, R, L, X, Y) =$
$\text{Deflate}(\widehat{\lambda}_1, \widehat{\mathbf{x}}_1, \widehat{\mathbf{y}}_1, \Lambda, R, L, \widehat{X}_{2:k}, \widehat{Y}_{2:k})$ {Alg. 5}
11: $p_{found} = p_{found} + 1$
12: Set $\widehat{\lambda}_1 = \widehat{\lambda}_2$, $k = k - 1$
13: **end if**
14: Set $k = k + 1$
15: Set the new pole estimate $s_{k+1} = \widehat{\lambda}_1$
16: **end while**

Selection Strategy

In step 8 of Alg. 2, the new pole estimate s_{k+1} has to be determined. A possible choice is to use the two-sided Rayleigh quotient (8) as it is used in DPA, but this choice does not take full advantage of subspace acceleration. Here, however, also another choice is possible, that is closer to the goal of computing the dominant poles.

Because in iteration k the interaction matrices $T \in \mathbb{C}^{k\times k}$ and $G \in \mathbb{C}^{k\times k}$ are of low order $k \ll n$ (see step 7 in Alg. 2), it is relatively cheap to compute the full eigendecomposition of the pencil (T, G). This provides k approximate eigentriplets $(\widehat{\lambda}_i, \widehat{\mathbf{x}}_i, \widehat{\mathbf{y}}_i)$. A natural thing to do is to choose the triplet $(\widehat{\lambda}_j, \widehat{\mathbf{x}}_j, \widehat{\mathbf{y}}_j)$ with the most dominant pole approximation: compute the corresponding residues $\widehat{R}_i = (\mathbf{c}^* \widehat{\mathbf{x}}_i)(\widehat{\mathbf{y}}_i^* \mathbf{b})$ of the k pairs and use the pole with the largest $|\widehat{R}_j|/|\text{Re}(\widehat{\lambda}_j)|$ as new estimate. Numerically, it is more robust to normalize $\widehat{\mathbf{x}}_i$ and $\widehat{\mathbf{y}}_i$ such that $\|\widehat{\mathbf{x}}_i\|_2 = \|\widehat{\mathbf{y}}_i\|_2 = 1$. Algorithm 3 orders the k approximate eigentriplets in decreasing dominance. The SADPA then continues with the new estimate $s_{k+1} = \widehat{\lambda}_1$.

Algorithm 3: $(\widehat{\Lambda}, \widehat{X}, \widehat{Y}) = \text{Sort}(T, G, X, Y, \mathbf{b}, \mathbf{c})$

INPUT: $T, G \in \mathbb{C}^{k\times k}$, $X, Y \in \mathbb{C}^{n\times k}$, $\mathbf{b}, \mathbf{c} \in \mathbb{C}^n$

OUTPUT: $\widehat{\Lambda} \in \mathbb{C}^n$, $\widehat{X}, \widehat{Y} \in \mathbb{C}^{n\times k}$ with $\widehat{\lambda}_1$ the pole approximation with largest scaled residue magnitude and $\widehat{\mathbf{x}}_1$ and $\widehat{\mathbf{y}}_1$ the corresponding approximate right and left eigenvectors

1: Compute eigentriplets of the pair (T, G):

$$(\widetilde{\lambda}_i, \widetilde{\mathbf{x}}_i, \widetilde{\mathbf{y}}_i), \quad i = 1, \ldots, k$$

2: Compute approximate eigentriplets of A as (with $\|\widehat{\mathbf{x}}_i\|_2 = \|\widehat{\mathbf{y}}_i\|_2 = 1$)

$$(\widehat{\lambda}_i = \widetilde{\lambda}_i, \widehat{\mathbf{x}}_i = X\widetilde{\mathbf{x}}_i, \widehat{\mathbf{y}}_i = Y\widetilde{\mathbf{y}}_i), \quad i = 1, \ldots, k$$

3: $\widehat{\Lambda} = [\widehat{\lambda}_1, \ldots, \widehat{\lambda}_k]$

4: $\widehat{X} = [\widehat{\mathbf{x}}_1, \ldots, \widehat{\mathbf{x}}_k]$

5: $\widehat{Y} = [\widehat{\mathbf{y}}_1, \ldots, \widehat{\mathbf{y}}_k]$

6: Compute residues $\widehat{R}_i = (\mathbf{c}^*\widehat{\mathbf{x}}_i)(\widehat{\mathbf{y}}_i^*\mathbf{b})$

7: Sort $\widehat{\Lambda}$, $\widehat{X}$, $\widehat{Y}$ in decreasing $|\widehat{R}_i|/|\text{Re}(\widehat{\lambda}_i)|$ order

Deflation

At the end of every iteration, in step 9, a convergence test is done as in DPA: if for the selected eigentriplet $(\widehat{\lambda}_j, \widehat{\mathbf{x}}_j, \widehat{\mathbf{y}}_j)$ the norm of the residual $\|A\widehat{\mathbf{x}}_j - \widehat{\lambda}_j\widehat{\mathbf{x}}_j\|_2$ is smaller than some tolerance ε, it is considered to be converged. In general more dominant eigentriplets are wanted and during the computation of the next eigentriplets, components in the direction of already found eigenvectors may enter the search spaces X and Y again. This may lead to repeated computation of the same eigentriplet. A well known technique to avoid repeated computation is deflation [18].

If already the right and left eigenvectors $\mathbf{x}_j$ and $\mathbf{y}_j$ are found, then it can be verified that, if the eigenvectors are exact, the matrix

$$\widetilde{A} = \prod_j (I - \frac{\mathbf{x}_j\mathbf{v}_j^*}{\mathbf{v}_j^*\mathbf{x}_j}) \cdot A \cdot \prod_j (I - \frac{\mathbf{x}_j\mathbf{v}_j^*}{\mathbf{v}_j^*\mathbf{x}_j})$$

has the same eigentriplets as A, but with the found eigenvalues transformed to zero (see also [5, 9]): let $\widehat{\mathbf{x}}$ be one of the k found exact right eigenvectors, i.e. $\widehat{\mathbf{x}} \in \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$. Then it follows from the orthogonality relations (see Sect. 2) that

$$\prod_j (I - \frac{\mathbf{x}_j\mathbf{y}_j^*}{\mathbf{y}_j^*\mathbf{x}_j}) \cdot \widehat{\mathbf{x}} = \widehat{\mathbf{x}} - \widehat{\mathbf{x}} = 0,$$

and hence $\widetilde{A}\widehat{\mathbf{x}} = 0$. On the other hand, let $\widehat{\mathbf{x}} \notin \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$ be a right eigenvector of A with eigenvalue $\widehat{\lambda}$. Then

$$\prod_j (I - \frac{\mathbf{x}_j\mathbf{y}_j^*}{\mathbf{y}_j^*\mathbf{x}_j}) \cdot \widehat{\mathbf{x}} = \widehat{\mathbf{x}},$$

and hence $\widetilde{A}\widehat{\mathbf{x}} = \widehat{\lambda}\widehat{\mathbf{x}}$. The result for left eigenvectors follows in a similar way. In finite arithmetic only *approximations* to exact eigentriplets are available and hence the computed eigenvalues are transformed to $\eta \approx 0$ (see also Sect. 3.3).

Using this, the space X needs to be orthogonally expanded with

$$\prod_j (I - \frac{\mathbf{x}_j \mathbf{y}_j^*}{\mathbf{y}_j^* \mathbf{x}_j}) \cdot \mathbf{x},$$

and similarly, the space V needs to be orthogonally expanded with

$$\prod_j (I - \frac{\mathbf{y}_j \mathbf{x}_j^*}{\mathbf{x}_j^* \mathbf{y}_j}) \cdot \mathbf{y}.$$

These projections are implemented with modified Gram-Schmidt (MGS) (see Alg. 4).

Algorithm 4: $X = \text{Expand}(X, R, L, \mathbf{x})$

INPUT: $X \in \mathbb{C}^{n \times k}$ with $X^*X = I$, $R, L \in \mathbb{C}^{n \times p}$, $\mathbf{x} \in \mathbb{C}^n$
OUTPUT: $X \in \mathbb{C}^{n \times (k+1)}$ with $X^*X = I$ and $\mathbf{x}_{k+1} = \prod_{j=1}^{p} (I - \frac{\mathbf{r}_j \mathbf{l}_j^*}{\mathbf{l}_j^* \mathbf{r}_j}) \cdot \mathbf{x}$

1: $\mathbf{x} = \prod_{j=1}^{p} (I - \frac{\mathbf{r}_j \mathbf{l}_j^*}{\mathbf{l}_j^* \mathbf{r}_j}) \cdot \mathbf{x}$
2: $\mathbf{x} = \text{MGS}(X, \mathbf{x})$
3: $X = [X, \mathbf{x}/\|\mathbf{x}\|_2]$

If a complex eigenvalue has converged, its complex conjugate is also a pole and the corresponding complex conjugate right and left eigenvectors can also be deflated. The complete deflation procedure is shown in Alg. 5.

Further Improvements and Remarks

It may happen that the subspaces X and Y become high-dimensional, especially when a large number of dominant poles is wanted. A common way to deal with this is to do an implicit restart [18]: if the subspaces X and Y reach a certain maximum dimension $k_{\max} \ll n$, they are reduced to a dimension $k_{\min} < k_{\max}$ by keeping the $k_{\min}$ most dominant approximate eigentriplets; the process is restarted with the reduced X and Y (already converged eigentriplets are not part of the active subspaces X and Y). This procedure is repeated until all poles are found.

The approximate residues $\widehat{R}_i$ can be computed without computing the approximate eigenvectors explicitly (step 2 and step 6 of Alg. 3): if the $\widetilde{\mathbf{x}}_i$ and $\widetilde{\mathbf{y}}_i$ are scaled so that $\|\widetilde{\mathbf{y}}_i\|_2 = \|\widetilde{\mathbf{x}}_i\| = 1$, then it follows that the $\widehat{R}_i$ can be computed as $\widehat{R}_i = ((\mathbf{c}^*X)\widetilde{\mathbf{x}}_i)(\widetilde{\mathbf{y}}_i^*(Y^*\mathbf{b}))$ $(= (\mathbf{c}^*\widehat{\mathbf{x}}_i)(\widehat{\mathbf{y}}_i^*\mathbf{b}))$.

Algorithm 5:
$(\Lambda, R, L, \widetilde{X}, \widetilde{Y}) = \text{Deflate}(\lambda, \mathbf{x}, \mathbf{v}, \Lambda, R, L, X, Y)$

INPUT: $\lambda \in \mathbb{C}$, $\mathbf{x}, \mathbf{y} \in \mathbb{C}^n$, $\Lambda \in \mathbb{C}^p$, $R, L \in \mathbb{C}^{n \times p}$, $X, Y \in \mathbb{C}^{n \times k}$

OUTPUT: $\Lambda \in \mathbb{C}^q$, $R, L \in \mathbb{C}^{n \times q}$,$\widetilde{X}, \widetilde{Y} \in \mathbb{C}^{n \times k}$, where $q = p + 1$ if λ has zero imaginary part and $q = p + 2$ if λ has nonzero imaginary part

1: $\Lambda = [\Lambda, \lambda]$
2: $R = [R, \mathbf{x}]$
3: $L = [L, \mathbf{y}]$
4: **if** imag$(\lambda) \neq 0$ **then**
5: {Also deflate complex conjugate}
6: $\Lambda = [\Lambda, \bar{\lambda}]$
7: $R = [R, \bar{\mathbf{x}}]$
8: $L = [L, \bar{\mathbf{y}}]$
9: **end if**
10: $\widetilde{X} = \widetilde{Y} = [\,]^{n \times 0}$
11: **for** $j = 1, \ldots, k$ **do**
12: $\widetilde{X} = \text{Expand}(\widetilde{X}, R, L, X_j)$
13: $\widetilde{Y} = \text{Expand}(\widetilde{Y}, L, R, Y_j)$
14: **end for**

Furthermore, as more eigentriplets have converged, approximations of new eigentriplets may become poorer or convergence may be hampered, due to rounding errors in the orthogonalization phase and the already converged eigentriplets. It is therefore advised to take a small tolerance $\varepsilon < 10^{-10}$. Besides that, as the estimate converges to a dominant pole, the right and left eigenvectors computed in step 3 and 4 of algorithm 2 are usually more accurate than the approximations computed in the selection procedure (although in exact arithmetic they are equal). In the deflation phase, it is therefore advised to take the most accurate of both.

Deflation can be implemented more efficiently in the following way: let $\mathbf{x}$ and $\mathbf{y}$ be right and left eigenvectors for eigenvalue λ and scaled such that $\mathbf{y}^*\mathbf{x} = 1$, with residue $R = (\mathbf{c}^*\mathbf{x})(\mathbf{y}^*\mathbf{b})$. With $\mathbf{b}_d = (I - \mathbf{x}\mathbf{y}^*)\mathbf{b}$ and $\mathbf{c}_d = (I - \mathbf{y}\mathbf{x}^*)\mathbf{c}$, it follows that the residue of λ in $H_d(s) = \mathbf{c}_d(sI - A)^{-1}\mathbf{b}_d$ is transformed to $R_d = 0$, while the residues of the remaining poles are left unchanged. Since $(sI - A)^{-1}\mathbf{b}_d \perp \mathbf{y}$ and $(sI - A)^{-*}\mathbf{c}_d \perp \mathbf{x}$, the orthogonalizations against found eigenvectors in step 5 and 6 of Alg. 2 are not needed any more (provided $\mathbf{b}$ and $\mathbf{c}$ are replaced by $\mathbf{b}_d$ and $\mathbf{c}_d$, respectively).

SADPA requires only one initial estimate. If rather accurate initial estimates are available, one can take advantage of this in SADPA by setting the next estimate after deflation to a new initial estimate (step 15 of Alg. 2).

Every iteration, two linear systems are to be solved (step 3 and 4). As was also mentioned in Sect. 3.2, this can be efficiently done by computing one LU-factorization and solving the systems by using L and U, and U^* and L^*, respectively. Because in practice the system matrix A is often very sparse, computation of the LU-factorization can be relatively inexpensive.

The selection criterion can easily be changed to another of the several existing indices of modal dominance [1, 7, 22]. Furthermore, the strategy can be restricted to considering only poles in a certain frequency range. Also, instead of providing the number of wanted poles, the procedure can be automated even further by providing the desired maximum error $|H(s) - H_k(s)|$ for a certain frequency range: the procedure continues computing new poles until the error bound is reached. Note that such an error bound requires that the transfer function of the complete model can be computed efficiently.

4 Generalizations

In this section, three variants of the dominant pole algorithm presented in the previous section are briefly discussed. Section 4.1 generalizes the theory to descriptor systems. In Sect. 4.2, the theory is extended to multi-input multi-ouput systems. A variant of DPA that computes the dominant zeros of a transfer function is described in Sect. 4.3.

4.1 Descriptor Systems

A more general representation of a dynamical system is

$$\begin{cases} E\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + \mathbf{b}u(t) \\ y(t) \quad = \mathbf{c}^*\mathbf{x}(t) + du(t), \end{cases} \tag{9}$$

where $A, E \in \mathbb{R}^{n\times n}$, $\mathbf{x}(t), \mathbf{b}, \mathbf{c} \in \mathbb{R}^n$ and $u(t), y(t), d \in \mathbb{R}$. The corresponding transfer function is

$$H(s) = \mathbf{c}^*(sE - A)^{-1}\mathbf{b} + d.$$

The case $E = I$ has been discussed already in Sect. 2. The descriptor system (9) arises for instance in electrical circuit simulation ($A = G$, $E = C$) and the sparse descriptor formulation of power systems (see for instance [15, 16]). The pencil (A, E) is assumed to be regular, that is, $A - \lambda E$ is singular only for a finite number of $\lambda \in \mathbb{C}$. If E is singular, system (9) is a system of differential-algebraic equations (DAE). If E is nonsingular, it is a system of ordinary differential equations (ODE).

The algorithms presented in this chapter can easily be adapted to handle (sparse) descriptor systems of the form (9). The changes essentially boil down to replacing I by E on most places and noting that for eigentriplets $(\lambda_j, \mathbf{x}_j, \mathbf{y}_j)$ with distinct finite λ_j, the relation $\mathbf{y}_i^* E\mathbf{x}_j = 0, i \neq j$ holds, and that for nondefective finite eigenvalues, the eigenvectors can be scaled so that $\mathbf{y}_i^* E\mathbf{x}_i = 1$. The modes corresponding to eigenvalues at infinity do not contribute to the effective transfer function behavior. For completeness, the changes are given for each algorithm:

- Algorithm 1:
  - Replace I by E in step 3 and 4.

  - Step 5 becomes
$$s_{k+1} = s_k - \frac{\mathbf{c}^*\mathbf{x} + d}{\mathbf{y}^* E \mathbf{x}}.$$
  - The criterion in step 6 becomes
$$\|A\mathbf{x} - s_{k+1}E\mathbf{x}\|_2 < \varepsilon.$$

- Algorithm 2:
  - Replace I by E in step 3 and 4.
  - Replace step 5 and 6 by
$$X = \text{Expand}(X, R, E^* \cdot L, \mathbf{x}),$$
$$Y = \text{Expand}(Y, L, E \cdot R, \mathbf{y}).$$
  - In step 7, use $G = Y^* E X$.
  - The criterion in step 9 becomes
$$\|A\widehat{\mathbf{x}}_1 - \widehat{\lambda}_1 E\widehat{\mathbf{x}}_1\|_2 < \varepsilon.$$

- Algorithm 5:
  - Replace step 12 and 13 by
$$\widetilde{X} = \text{Expand}(\widetilde{X}, R, E^* \cdot L, X_j),$$
$$\widetilde{Y} = \text{Expand}(\widetilde{Y}, L, E \cdot R, Y_j).$$

4.2 MIMO Systems

For a multi-input multi-output (MIMO) system
$$\begin{cases} E\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t) \\ \mathbf{y}(t) \quad = C^*\mathbf{x}(t) + D\mathbf{u}(t), \end{cases}$$
where $A, E \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{n\times p}$, $\mathbf{x}(t) \in \mathbb{R}^n$, $\mathbf{u}(t) \in \mathbb{R}^m$, $\mathbf{y}(t) \in \mathbb{R}^p$ and $D \in \mathbb{R}^{p\times m}$, the transfer function $H(s) : \mathbb{C} \longrightarrow \mathbb{C}^{p\times m}$ is defined as
$$H(s) = C^*(sE - A)^{-1}B + D. \tag{10}$$
The dominant poles of (10) are those $s \in \mathbb{C}$ for which $\sigma_{\max}(H(s)) \to \infty$. For square transfer functions ($m = p$), there is an equivalent criterion: the dominant poles are those $s \in \mathbb{C}$ for which $\lambda_{\min}(H^{-1}(s)) \to 0$. This leads, for square transfer functions, to the following Newton scheme:
$$s_{k+1} = s_k - \frac{1}{\mu_{\min}} \frac{1}{\mathbf{v}^* C^* (s_k E - A)^{-2} B \mathbf{u}},$$
where $(\mu_{\min}, \mathbf{u}, \mathbf{v})$ is the eigentriplet of $H^{-1}(s_k)$ corresponding to $\lambda_{\min}(H^{-1}(s_k))$. An algorithm for computing the dominant poles of a MIMO transfer function can be readily derived from Alg. 1. The reader is referred to [13] for the initial MIMO DPA algorithm and to [15] for an algorithm similar to SADPA, generalizations to non-square MIMO systems and more details.

4.3 Computing Zeros of a Transfer Function

The zeros of a transfer function $H(s) = \mathbf{c}^*(sE - A)^{-1}\mathbf{b} + d$ are those $s \in \mathbb{C}$ for which $H(s) = 0$. An algorithm, very similar to Alg. 1, can be derived by noting that a Newton scheme for computing the zeros of a transfer function is given by

$$s_{k+1} = s_k + \frac{\mathbf{c}^*(s_k E - A)^{-1}\mathbf{b} + d}{\mathbf{c}^*(s_k E - A)^{-2}\mathbf{b}}.$$

A slightly different formulation can be found in [12].

5 Numerical Examples

5.1 A Small Test System

For illustrational purposes, SADPA was applied to a transfer function of the New England test system, a model of a power system. This small benchmark system has 66 state variables (for more information, see [11]). The tolerance used was $\varepsilon = 10^{-10}$ and no restarts were used. Every iteration, the pole approximation $\widehat{\lambda}_j$ with largest $|\widehat{R}_j|/|\mathrm{Re}(\widehat{\lambda}_j)|$ was selected. Table 1 shows the found dominant poles and the iteration number in which the pole converged. Bodeplots of two modal equivalents are shown in Fig. 1 and Fig. 2. The quality of the modal equivalent increases with the number of found poles, as can be observed from the better match of the exact and reduced transfer function.

5.2 A Large-Scale Descriptor System

The Brazilian Interconnected Power System (BIPS) is a year 1999 planning model that has been used in practice (see [16] for more technical details). The size of the sparse matrices A and E is $n = 13,251$ (the number of states in the dense state space realization is $1,664$). The corresponding transfer function has a non-zero direct transmission term d. Figure 3 shows the frequency response of the complete model and the reduced model (41 states) together with the error. Both the magnitude and the phase plot show good matches of the exact and the reduced transfer functions (a relative error of approximately $\|H(s) - H_k(s)\|/\|H_k(s)\| = 0.1$, also for the

Table 1. Results for SADPA applied to the New England test system ($s_1 = 1i$).

#poles	#states	new pole	iteration	Bodeplot
1	2	$-0.4672 \pm 8.9644i$	13	-
2	4	$-0.2968 \pm 6.9562i$	18	-
3	5	-0.0649	21	Fig. 1
4	7	$-0.2491 \pm 3.6862i$	25	-
5	9	$-0.1118 \pm 7.0950i$	26	-
6	11	$-0.3704 \pm 8.6111i$	27	Fig. 2

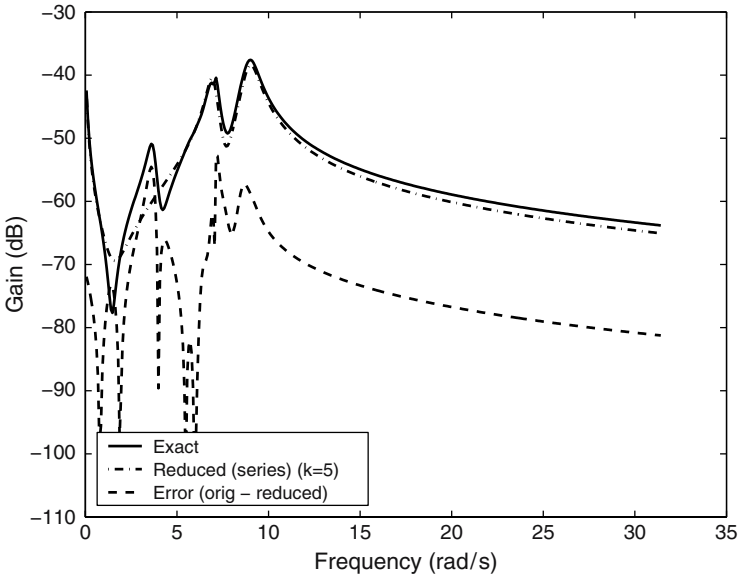

Fig. 1. Bode plot of modal equivalent, complete model and error for the transfer function of the New England test system (5 states in the modal equivalent, 66 in the complete model).

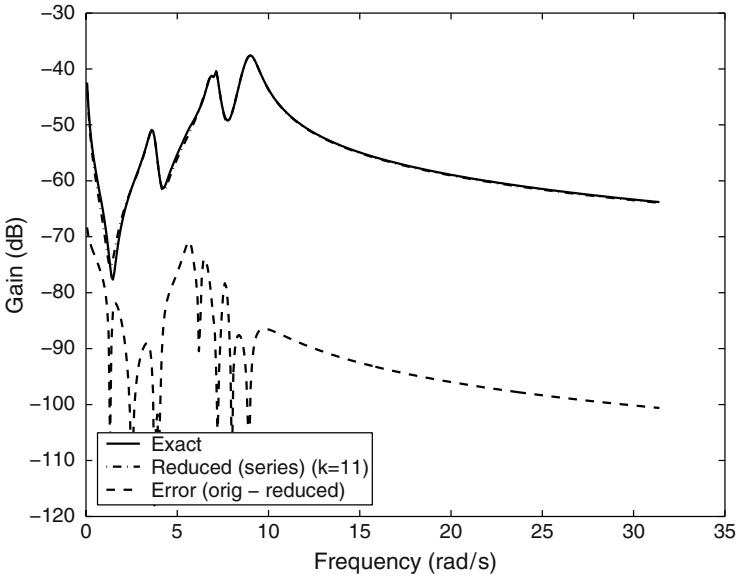

Fig. 2. Bode plot of modal equivalent, complete model and error for the transfer function of the New England test system (11 states in the modal equivalent, 66 in the complete model).

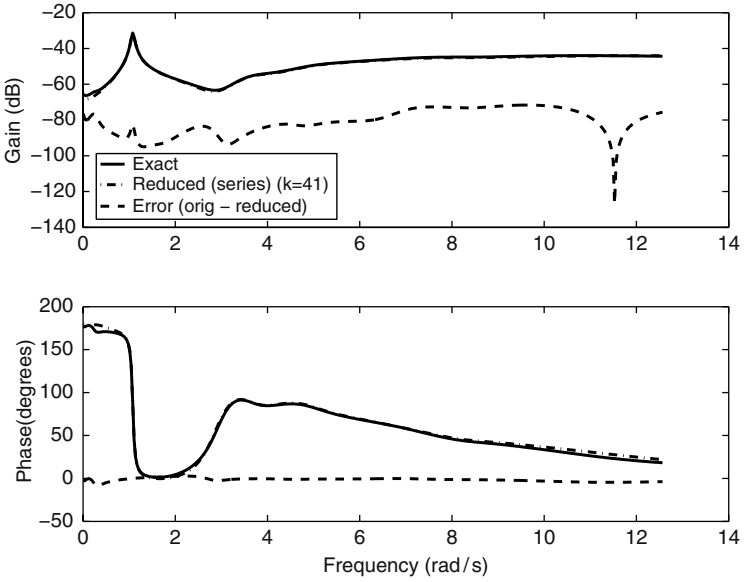

Fig. 3. Bode plot of modal equivalent, complete model and error for transfer function $P_{sc}(s)/B_{sc}(s)$ of BIPS (41 in the modal equivalent, 1664 in the complete model).

DC-gain $H(0)$). Figure 4 shows the corresponding step response (step $u = 0.01$)[1]. The reduced model nicely captures the system oscillations. The reduced model (30 poles, 56 states) was computed by SADPA in 341 LU-factorizations ($k_{\min} = 1$, $k_{\max} = 10$). This reduced model could be reduced further to 41 states (22 poles) by removing less dominant contributions, without decreasing the quality of the reduced model much.

5.3 A PEEC Example

This descriptor model arises from a partial element equivalent circuit (PEEC) of a patch antenna structure and the dimension of the matrices A and E is $n = 480$ (see [4, 17, 23] for more details and the model data). The system is known as a difficult problem, because it has many poles close to each other [8]. Figure 5 shows the Bodeplot of the complete model and the reduced model (45 poles, 89 states) computed by SADPA (initial estimate $s_1 = 100i$, $k_{\min} = 5$, $k_{\max} = 15$). The approximation is almost exact for the frequency range $[0.1, \ldots, 10^2]$ rad/sec. For frequencies higher than 10^2 rad/sec, the quality is less good.

[1] If $h_k(t)$ is the inverse Laplace transform of $H_k(s)$ (3), the step response for step $u(t) = c$ of the reduced model is given by $y(t) = \int_0^t h(t)u(t) = c(\sum_{i=1}^{k}(\frac{R_i}{\lambda_i}(\exp(\lambda_i t) - 1)) + d)$.

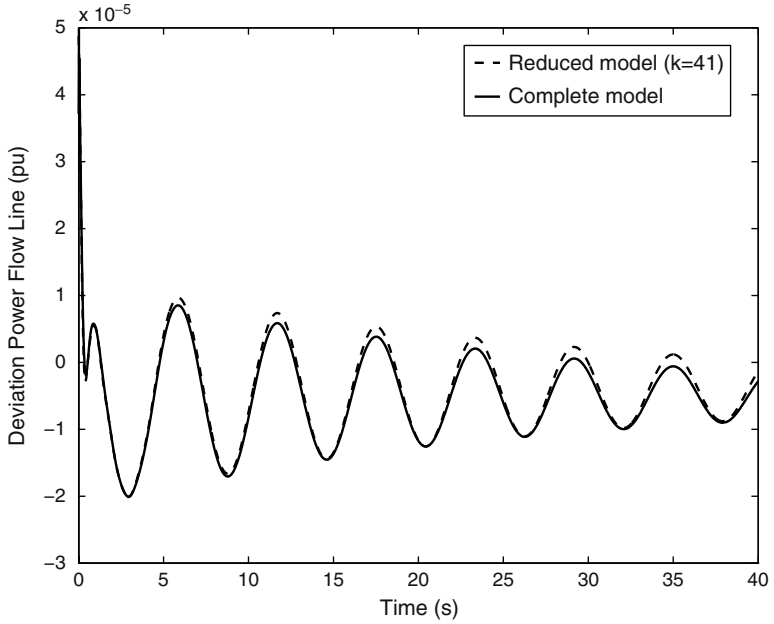

Fig. 4. Step responses for transfer function $P_{sc}(s)/B_{sc}(s)$ of BIPS, complete model and modal equivalent (41 states in the modal equivalent, 1664 in the complete model, step disturbance of 0.01 pu).

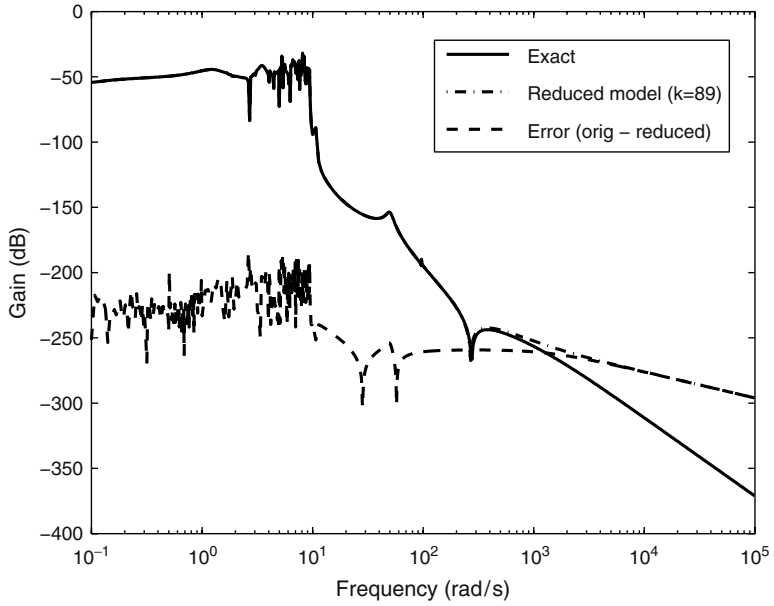

Fig. 5. Bode plot of modal equivalent, complete model and error for transfer function of the PEEC model (89 states in the modal equivalent, 480 in the complete model).

6 Conclusions

The algorithms presented in this chapter are efficient, automatic methods to compute dominant poles of large-scale transfer functions. It has been shown how the corresponding left and right eigenvectors can be used to construct a reduced order model, also known as the modal equivalent, of the original system. Although the methods may not be successful for every system, the numerical results for real-life systems and benchmarks indicate that the methods are applicable to a large class of systems.

Acknowledgement

Henk van der Vorst, Gerard Sleijpen and Nelson Martins are thanked for their valuable comments to earlier versions of this chapter.

References

1. AGUIRRE, L. A. Quantitative Measure of Modal Dominance for Continuous Systems. In *Proc. of the 32nd Conference on Decision and Control* (December 1993), pp. 2405–2410.
2. ANTOULAS, A. C. *Approximation of Large-Scale Dynamical Systems*. SIAM, 2005.
3. BEZERRA, L. H. Written discussion to [11]. *IEEE Trans. Power Syst. 11*, 1 (Feb. 1996), 168.
4. CHAHLAOUI, Y., AND VAN DOOREN, P. A collection of Benchmark examples for model reduction of linear time invariant dynamical systems. SLICOT Working Note 2002-2, 2002.
5. FOKKEMA, D. R., SLEIJPEN, G. L. G., AND VAN DER VORST, H. A. Jacobi-Davidson style QR and QZ algorithms for the reduction of matrix pencils. *SIAM J. Sc. Comp. 20*, 1 (1998), 94–125.
6. GOLUB, G. H., AND VAN LOAN, C. F. *Matrix Computations*, third ed. John Hopkins University Press, 1996.
7. GREEN, M., AND LIMEBEER, D. J. N. *Linear Robust Control*. Prentice-Hall, 1995.
8. HERES, P. J. *Robust and efficient Krylov subspace methods for model order reduction*. PhD thesis, Eindhoven University of Technology, 2005.
9. HOCHSTENBACH, M. E., AND SLEIJPEN, G. L. G. Two-sided and alternating Jacobi-Davidson. *Lin. Alg. Appl. 358*, 1-3 (2003), 145–172.
10. KAILATH, T. *Linear Systems*. Prentice-Hall, 1980.
11. MARTINS, N., LIMA, L. T. G., AND PINTO, H. J. C. P. Computing dominant poles of power system transfer functions. *IEEE Trans. Power Syst. 11*, 1 (Feb. 1996), 162–170.
12. MARTINS, N., PINTO, H. J. C. P., AND LIMA, L. T. G. Efficient methods for finding transfer function zeros of power systems. *IEEE Trans. Power Syst. 7*, 3 (Aug. 1992), 1350–1361.
13. MARTINS, N., AND QUINTÃO, P. E. M. Computing dominant poles of power system multivariable transfer functions. *IEEE Trans. Power Syst. 18*, 1 (Feb. 2003), 152–159.
14. PARLETT, B. N. The Rayleigh quotient iteration and some generalizations for nonnormal matrices. *Math. Comp. 28*, 127 (July 1974), 679–693.

15. ROMMES, J., AND MARTINS, N. Efficient computation of multivariable transfer function dominant poles using subspace acceleration. *IEEE Trans. Power Syst. 21*, 4 (Nov. 2006), 1471–1483.
16. ROMMES, J., AND MARTINS, N. Efficient computation of transfer function dominant poles using subspace acceleration. *IEEE Trans. Power Syst. 21*, 3 (Aug. 2006), 1218–1226.
17. RUEHLI, A. E. Equivalent circuit models for three dimensional multi-conductor systems. *IEEE Trans. Microwave Theory Tech. 22* (Mar. 1974), 216–221.
18. SAAD, Y. *Numerical Methods for Large Eigenvalue Problems: Theory and Algorithms*. Manchester University Press, 1992.
19. SLEIJPEN, G. L. G., AND VAN DER VORST, H. A. A Jacobi-Davidson iteration method for linear eigenvalue problems. *SIAM J. Matrix Anal. Appl. 17*, 2 (1996), 401–425.
20. SMITH, J. R., HAUER, J. F., TRUDNOWSKI, D. J., FATEHI, F., AND WOODS, C. S. Transfer function identification in power system application. *IEEE Trans. Power Syst. 8*, 3 (Aug. 1993), 1282–1290.
21. THE MATHWORKS, INC. Matlab.
22. VARGA, A. Enhanced modal approach for model reduction. *Math. Mod. Syst.*, 1 (1995), 91–105.
23. VERBEEK, M. E. Partial element equivalent circuit (PEEC) models for on-chip passives and interconnects. *Int. J. Num. Mod.: Electronic Networks, Devices and Fields 17*, 1 (Jan. 2004), 61–84.