Chapter 3

Subspace accelerated DPA and Jacobi-Davidson style methods

Abstract. This chapter describes a new algorithm for the computation of the dominant poles of a high-order scalar transfer function. The algorithm, called the Subspace Accelerated Dominant Pole Algorithm (SADPA), is able to compute the full set of dominant poles and to produce good modal equivalents automatically, without any human interaction. It is shown that SADPA is equivalent to two-sided Jacobi-Davidson, if the (correction) equations for both are solved exactly. The effects of subspace acceleration on local and global convergence are studied. The modal approximations computed by SADPA are compared to reduced-order models computed by rational Krylov methods, and it is shown how both methods can be applied to improve each other.

Key words. small-signal stability, poorly-damped oscillations, power system dynamics, transfer function, system poles, model reduction, dominant pole spectrum, large-scale systems, sparse eigenanalysis, modal equivalents, two-sided Jacobi-Davidson

3.1 Introduction

Recent work on power system stability, controller design and electromagnetic transients has used several advanced model reduction techniques [29, 108, 119, 134, 153], that produce good results but impose high computational costs. Modal model reduction is a cost-effective alternative for large-scale systems, when only a fraction of the system pole spectrum is controllable-observable for the transfer function of interest. The concept of pole dominance, as utilized in this chapter, implies high controllability and observability in the chosen transfer function. Modal reduction produces transfer function modal equivalents from the knowledge of the dominant poles and their corresponding residues, but requires specialized eigensolution methods. Existing methods are still not capable enough to produce automatically the full set of truly dominant poles [90, 91, 93] for large system models. A good sur-

vey on model reduction methods employing either singular value decompositions or moment matching based methods can be found in [5, 6]. A good introduction on modal model reduction on state-space models can be found in [68].

In this chapter, a new variant of the Dominant Pole Algorithm (DPA) [91] and the Dominant Pole Spectrum Eigensolver (DPSE) [90] will be proposed: Subspace Accelerated DPA (SADPA). Instead of computing the dominant poles of a scalar transfer function simultaneously, the dominant poles and corresponding residues are computed one by one by selecting the most dominant approximation every iteration. This approach leads to a faster, more robust and more flexible algorithm. To avoid repeated computation of the same dominant poles, a deflation strategy is used. SADPA directly operates on implicit state-space systems, also known as descriptor systems, which are very sparse in practical power system applications.

The subspace accelerated dominant pole algorithm SADPA (originally presented in [123], see also the Addendum of this chapter) will be related to two-sided Jacobi-Davidson [74]. It will be shown that if the linear systems that arise in SADPA, and the correction equations that arise in two-sided JD, are solved exactly, and if the same Ritz value selection criterion is used, then both methods are equivalent and compute the same dominant poles. This may be surprising at first sight, since *without* subspace acceleration, DPA and two-sided JD in general do not converge to the same pole: it was shown in Chapter 2 that DPA tends to converge to the most dominant pole in the neighborhood of the initial guess, while RQI tends to converge to the pole closest to the initial guess. Subspace acceleration in combination with a proper selection criterion makes both methods equivalent and hence enables two-sided Jacobi-Davidson to converge to dominant poles.

SADPA is a generalization of DPA to compute more than one dominant pole, by using deflation, subspace acceleration, and a proper selection strategy. Deflation is needed to avoid repeated computation of already found poles and hence can be seen as a necessary ingredient. Subspace acceleration, on the other hand, has the strategic advantage of better global convergence, but the computational drawback of having to keep orthogonal search spaces. If deflation is organized as deflation by restriction [112, Sect. 5.2], then DPA can easily and efficiently be extended to a multi-pole algorithm, without subspace acceleration. Practically speaking, however, this approach has some disadvantages, and subspace acceleration is an effective way to deal with these difficulties.

In the large-scale setting it is not always feasible to solve the linear systems and correction equations exactly. For Jacobi-Davidson style methods, it is well known that it is sufficient for local convergence to solve the correction equation up to a certain accuracy [51, 140]. In SADPA, however, the use of inexact solves may cause stagnation, even if subspace acceleration is used. This behavior is also observed for inexact (subspace accelerated) Rayleigh quotient iteration [74, 154]. Motivated by this observation and other well known arguments, two-sided Jacobi-Davidson is to be preferred if only inexact solves are feasible.

Besides presenting subspace accelerated DPA, the goal of this chapter is threefold. Firstly, is shown that SADPA and two-sided Jacobi-Davidson are equivalent methods if the equations are solved exactly. Secondly, the role of subspace ac-

celeration (and a proper selection criterion) with respect to global convergence is analyzed, and the influence of subspace acceleration on the asymptotic rate of convergence in general is studied. Thirdly, it will be shown that two-sided Jacobi-Davidson is an effective method to compute dominant poles if only inexact solves are feasible. All results are illustrated by numerical experiments.

The chapter is organized as follows. Section 3.2 gives definitions and properties of transfer functions and dominant poles, and describes the basic DPA algorithm. In Section 3.3 it is described how DPA can be extended to subspace accelerated DPA (SADPA), to compute more than one dominant pole, and in Section 3.4 it is shown how two-sided Jacobi-Davidson can be formulated to become equivalent to SADPA. The role of subspace acceleration with respect to global and local convergence is analyzed in Section 3.5. Inexact variants of SADPA and two-sided JD are discussed in Section 3.6. Numerical results are presented in Section 3.7. In Section 3.8 SADPA is compared to rational Krylov based model reduction methods. Section 3.9 concludes.

3.2 Transfer functions and dominant poles

In this chapter, the dynamical systems $(E, A, \mathbf{b}, \mathbf{c}, d)$ are of the form

$$\left\{ \begin{array}{rcl} E\dot{\mathbf{x}}(t) & = & A\mathbf{x}(t) + \mathbf{b}u(t) \\ y(t) & = & \mathbf{c}^*\mathbf{x}(t) + du(t), \end{array} \right. \tag{3.2.1}$$

where $A, E \in \mathbb{R}^{n\times n}$, E may be singular, $\mathbf{b}, \mathbf{c}, \mathbf{x}(t) \in \mathbb{R}^n$, $u(t), y(t), d \in \mathbb{R}$. The vectors $\mathbf{b}$ and $\mathbf{c}$ are called the input, and output map, respectively. The transfer function $H : \mathbb{C} \longrightarrow \mathbb{C}$ of (3.2.1) is defined as

$$H(s) = \mathbf{c}^*(sE - A)^{-1}\mathbf{b} + d. \tag{3.2.2}$$

The poles of transfer function (3.2.2) are a subset of the eigenvalues $\lambda_i \in \mathbb{C}$ of the matrix pencil (A, E). An eigentriplet $(\lambda_i, \mathbf{x}_i, \mathbf{y}_i)$ is composed of an eigenvalue λ_i of (A, E) and corresponding right and left eigenvectors $\mathbf{x}_i, \mathbf{y}_i \in \mathbb{C}^n$ (identified by their components in $\mathbf{b}$ and $\mathbf{c}$):

$$\begin{array}{rcll} A\mathbf{x}_i & = & \lambda_i E\mathbf{x}_i, & \mathbf{x}_i \neq 0, \\ \mathbf{y}_i^*A & = & \lambda_i \mathbf{y}_i^*E, & \mathbf{y}_i \neq 0, \qquad (i = 1, \ldots, n). \end{array}$$

Assuming that the pencil is nondefective, the right and left eigenvectors corresponding to finite eigenvalues can be scaled so that $\mathbf{y}_i^*E\mathbf{x}_i = 1$. Furthermore, it is well known that left and right eigenvectors corresponding to distinct eigenvalues are E-orthogonal: $\mathbf{y}_i^*E\mathbf{x}_j = 0$ for $i \neq j$. The transfer function $H(s)$ can be expressed as a sum of residues $R_i \in \mathbb{C}$ over the $\tilde{n} \leq n$ finite first order poles [78]:

$$H(s) = \sum_{i=1}^{\tilde{n}} \frac{R_i}{s - \lambda_i} + R_\infty + d, \tag{3.2.3}$$

where the residues R_i are

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

and R_∞ is the constant contribution of the poles at infinity (often zero).

Although there are different indices of modal dominance [4, 68, 159], the following will be used in this chapter.

Definition 3.2.1. *A pole λ_i of $H(s)$ with corresponding right and left eigenvectors $\mathbf{x}_i$ and $\mathbf{y}_i$ ($\mathbf{y}_i^*E\mathbf{x}_i = 1$) is called* the *dominant pole if $|R_i|/|Re(\lambda_i)| > |R_j|/|Re(\lambda_j)|$, for all $j \neq i$.*

More generally, a pole λ_i is called dominant if $|R_i|/|\mathrm{Re}(\lambda_i)|$ is not small compared to $|R_j|/|\mathrm{Re}(\lambda_j)|$, for all $j \neq i$. A dominant pole is well observable and controllable in the transfer function. This can also be seen in the corresponding Bode-plot, which is a plot of $|H(i\omega)|$ against $\omega \in \mathbb{R}$: peaks occur at frequencies ω close to the imaginary parts of the dominant poles of $H(s)$. An approximation of $H(s)$ that consists of $k < n$ terms with $|R_j|/|\mathrm{Re}(\lambda_j)|$ above some value, determines the effective transfer function behavior [144] and is also known as transfer function modal equivalent:

Definition 3.2.2. *A transfer function modal equivalent $H_k(s)$ is an approximation of a transfer function $H(s)$ that consists of $k < n$ terms:*

$$H_k(s) = \sum_{j=1}^{k} \frac{R_j}{s - \lambda_j} + d. \tag{3.2.4}$$

A modal equivalent that consists of the most dominant terms determines the effective transfer function behavior [144]. 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, E) as columns, such that $Y^*AX = \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_k)$, with $Y^*EX = I$, then the corresponding (complex) reduced system follows by setting $\mathbf{x} = X\tilde{\mathbf{x}}$ and multiplying from the left by Y^*:

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

In practice, it is advisable to make a real reduced model in the following way: for every complex pole triplet $(\lambda, \mathbf{x}, \mathbf{y})$, construct real bases for the right and left eigenspace via $[\mathrm{Re}(\mathbf{x}), \mathrm{Im}(\mathbf{x})]$ and $[\mathrm{Re}(\mathbf{y}), \mathrm{Im}(\mathbf{y})]$, respectively. Let the columns of X_r and Y_r be such bases, respectively. Because the complex conjugate eigenvectors are also in this space, the real bases for the eigenspaces are still (at most) k dimensional. The real reduced model can be formed by using X_r and Y_r in $(Y_r^*EX_r, Y_r^*AX_r, Y_r^*\mathbf{b}, X_r^*\mathbf{c}, d)$.

The dominant poles are specific (complex) eigenvalues of the pencil (A, E) and usually form a small subset of the spectrum of (A, E), so that rather accurate modal equivalents may be possible for $k \ll n$. The dominant poles can be located anywhere in the spectrum. Since the dominance of a pole is independent of d, without loss of generality $d = 0$ in the following.

3.2.1 The Dominant Pole Algorithm (DPA)

The poles of transfer function (3.2.2) are the $\lambda \in \mathbb{C}$ for which $\lim_{s\to\lambda} |H(s)| = \infty$ and, as was described in Section 2.3, can be computed via the roots of $G(s) = 1/H(s)$. Applying Newton's method leads to the scheme

$$s_{k+1} = s_k - \frac{\mathbf{c}^* \mathbf{v}_k}{\mathbf{w}_k^* E \mathbf{v}_k}, \tag{3.2.5}$$

where $\mathbf{v}_k = (s_k E - A)^{-1}\mathbf{b}$ and $\mathbf{w}_k = (s_k E - A)^{-*}\mathbf{c}$. The algorithm based on this scheme, also known as the Dominant Pole Algorithm (DPA) [91], is shown in Algorithm 2.1 in Chapter 2. Note that strictly speaking the definition of dominance used here is Definition 2.2.1.

The two linear systems that need to be solved in step 3 and 4 of Algorithm 2.1 to compute the Newton update in (3.2.5) can be efficiently solved using one LU-factorization $LU = s_k E - A$, by noting that $U^* L^* = (s_k E - A)^*$. If an exact LU-factorization is not available, one has to use inexact Newton schemes, such as inexact Rayleigh Quotient Iteration and Jacobi-Davidson style methods [74, 140, 148], a topic that will be studied in more detail in Section 3.6. In the next section, extensions of DPA are presented that are able to compute more than one eigenvalue in an effective and efficient way.

3.3 Deflation and subspace acceleration

In Chapter 2 the convergence of DPA is analyzed and compared to the convergence of two-sided Rayleigh quotient iteration. For symmetric matrices, it is shown that the convergence neighborhood of the dominant pole is larger for DPA than for RQI, and numerical experiments confirm that this is also true for general descriptor systems. DPA also has better global convergence to the dominant poles than two-sided RQI. The price one has to pay for this is that the asymptotic rate of convergence is quadratic, against the cubic rate of convergence of RQI. In practice, however, the difference is hardly noticeable, since the number of additional iterations is often only 1 or 2.

The situation may be different if deflation and subspace acceleration are used, and selection strategies are required to select a Ritz value at every iteration. In practical applications often more than one dominant pole is wanted: one is interested in all the dominant poles, no matter what definition of dominance is used. Simply running the single pole algorithm DPA for a number of different initial shifts will most likely result in duplicate dominant poles. In [90] the Dominant Pole Spectrum Eigensolver (DPSE) was presented as a block variant of DPA to compute multiple dominant poles, by running k DPA iterations in parallel for k different shifts. Every iteration, this leads to k approximate right and left eigenvectors $\mathbf{v}^{(i)}$ and $\mathbf{w}^{(i)}$ $(i = 1, \ldots, k)$. The shifts for the new iteration are the eigenvalues of the projected pencil (W^*AV, W^*EV), where the columns of V and W are the approximate right and left eigenvectors $\mathbf{v}^{(i)}$ and $\mathbf{w}^{(i)}$, respectively. DPSE, however, requires blocks of size $n \times k$, where k is the number of wanted poles, and this

may be impractical when k is large, since it leads to projected pencils of order k. Besides that, there may be simultaneous convergence to complex conjugate poles (redundancy), and, like in DPA, components in the direction of other eigenvectors are discarded at the end of every iteration.

A well known strategy to avoid computation of already found eigenvalues is deflation, see for instance [132]. It is also known that subspace acceleration may improve the global convergence: for an $n \times n$ problem, the subspace accelerated algorithm converges within at most n iterations, although in practice it may need only $k \ll n$ iterations and will almost never build a full search space of dimension n, but restart every $k_{max} \ll n$ iterations. The use of subspace acceleration requires that every iteration an approximate pole has to be selected from the available approximations. This also may improve the global convergence, since better approximations than the initial estimate, which may be a rather crude approximation, become available during the process. The effect on the local convergence, however, is less understood. The role of subspace acceleration will be analyzed in more detail in Section 3.5.

In Section 3.3.1, first a variant of DPA for the computation of more than one pole that does *not* use subspace acceleration is discussed. Although this variant can be implemented very efficiently with only constant costs for deflation, it also has some disadvantages that may make it of less use in practice. The problems that arise in this variant are an additional motivation for the use of subspace acceleration in Section 3.3.2. In Section 3.4, a variant based on two-sided Jacobi-Davidson is presented, and it is shown that subspace accelerated DPA (SADPA) and two-sided Jacobi-Davidson are equivalent methods.

Throughout the rest of this chapter, let the $(n \times k)$ matrices X_k and Y_k have as their columns the normalized (found) right and left eigenvectors $\mathbf{x}_i$ and $\mathbf{y}_i$ $(i = 1, \ldots, k)$ of (A, E), respectively, and let Λ_k be a diagonal $(k \times k)$ matrix with on its diagonal the corresponding eigenvalues, i.e. $\Lambda_k = \mathrm{diag}(\lambda_1, \ldots, \lambda_k)$, $Y_k^* A X_k = \Lambda_k$ and $Y_k^* E X_k = I$. For ease of notation, the subscript k will be omitted if this does not lead to confusion.

3.3.1 DPA with deflation by restriction

It can be verified that if $X \equiv X_k$ and $Y \equiv Y_k$ have as their columns exact eigenvectors (normalized so that $Y^* E X = I$), then the system $(E_d, A_d, \mathbf{b}_d, \mathbf{c}_d)$, where

$$\begin{aligned} E_d &= (I - EXY^*)E(I - XY^*E), \\ A_d &= (I - EXY^*)A(I - XY^*E), \\ \mathbf{b}_d &= (I - EXY^*)\mathbf{b}, \\ \mathbf{c}_d &= (I - E^*YX^*)\mathbf{c}, \end{aligned}$$

has the same poles, eigenvectors and residues, but with the found λ_i $(i = 1, \ldots, k)$ and corresponding R_i transformed to 0. So in order to avoid recomputing the same pole, DPA could be applied to the deflated system $(E_d, A_d, \mathbf{b}_d, \mathbf{c}_d)$ after having found one or more poles. This would require solves with $(sE_d - A_d)$ and $(sE_d - A_d)^*$

in step 4 and 5 of Algorithm 3.1[1], but the following theorem shows that it is sufficient to only replace $\mathbf{b}$ by $\mathbf{b}_d$ and $\mathbf{c}$ by $\mathbf{c}_d$ to ensure deflation.

Theorem 3.3.1. *The deflated transfer function* $H_d(s) = \mathbf{c}_d^*(sE - A)^{-1}\mathbf{b}_d$, *where*

$$\mathbf{b}_d = (I - EXY^*)\mathbf{b} \quad \text{and} \quad \mathbf{c}_d = (I - E^*YX^*)\mathbf{c},$$

has the same poles λ_i *and corresponding residues* R_i *as* $H(s) = \mathbf{c}^*(sE-A)^{-1}\mathbf{b}$, *but with the residues* R_i *corresponding to the found poles* λ_i $(i = 1, \ldots, k)$ *transformed to* $R_i = 0$.

Proof. Recall that the residue corresponding to an eigentriplet $(\lambda_i, \mathbf{x}_i, \mathbf{y}_i)$ of (A, E) is defined as $R_i = (\mathbf{c}^*\mathbf{x}_i)(\mathbf{y}_i^*\mathbf{b})$. Let $(\lambda, \mathbf{x}, \mathbf{y})$ be a found eigentriplet with $R = (\mathbf{c}^*\mathbf{x})(\mathbf{y}^*\mathbf{b})$ and hence $X(Y^*E\mathbf{x}) = \mathbf{x}$ and $(\mathbf{y}^*EX)Y^* = \mathbf{y}^*$. Note that clearly $Y^*\mathbf{b}_d = X^*\mathbf{c}_d = 0$, and it follows immediately that the corresponding residue in $H_d(s)$ is

$$R_d = (\mathbf{c}_d^*\mathbf{x})(\mathbf{y}^*\mathbf{b}_d) = (\mathbf{c}^*(I - XY^*E)\mathbf{x})(\mathbf{y}^*(I - EXY^*)\mathbf{b}) = 0.$$

Similarly, if $(\lambda, \mathbf{x}, \mathbf{y})$ is not deflated from $H(s)$ and hence $Y^*E\mathbf{x} = 0$ and $\mathbf{y}^*EX = 0$, then the corresponding residue is

$$R_d = (\mathbf{c}_d^*\mathbf{x})(\mathbf{y}^*\mathbf{b}_d) = (\mathbf{c}^*(I - XY^*E)\mathbf{x})(\mathbf{y}^*(I - EXY^*)\mathbf{b}) = (\mathbf{c}^*\mathbf{x})(\mathbf{y}^*\mathbf{b}) = R.$$

Since A and E are left unchanged, so are the eigenvalues of (A, E) and hence the poles. □

In other words, by using $\mathbf{b}_d$ and $\mathbf{c}_d$ the found dominant poles are degraded to non dominant poles of $H_d(s)$, while not changing the dominance of the remaining poles. Graphically, the peaks caused by the found poles are 'flattened' in the Bode plot (see also Figure 3.1).

Corollary 3.3.2. *Since the residues of the found poles are transformed to zero, the found poles are no longer dominant poles of* $H_d(s)$. *Hence these poles will not be recomputed by DPA applied to* $H_d(s)$.

Note that if $H(s) = \mathbf{c}^*(sE-A)^{-1}\mathbf{b}+d$ with $d = 0$, then the deflated poles in fact become zeros of $H_d(s)$. Finally, the following theorem shows that DPA applied to $H_d(s) = \mathbf{c}_d^*(sE - A)^{-1}\mathbf{b}_d$ and DPA applied to $H_{\tilde{d}}(s) = \mathbf{c}_d^*(sE_d - A_d)^{-1}\mathbf{b}_d$ produce the same results.

Theorem 3.3.3. *Given an initial shift* s_0, *DPA(*E, A, $\mathbf{b}_d$, $\mathbf{c}_d$, s_0*) and DPA(*E_d, A_d, $\mathbf{b}_d$, $\mathbf{c}_d$, s_0*) produce, in exact arithmetic, the same iterates and results.*

Proof. Denote the pole estimates produced by DPA(E, A, $\mathbf{b}_d$, $\mathbf{c}_d$, s_0) by s_k, and the pole estimates produced by DPA(E_d, A_d, $\mathbf{b}_d$, $\mathbf{c}_d$, s_0) by $\tilde{s}_k$. Let $\mathbf{v} = (s_kE - A)^{-1}\mathbf{b}_d$ and $\mathbf{w} = (s_kE - A)^{-*}\mathbf{c}_d$. It follows that $Y^*E\mathbf{v} = Y^*E(s_kE - A)^{-1}\mathbf{b}_d =$

[1] Note that $(sE_d - A_d)$ would never be computed explicitly, and that systems $(sE_d - A_d)\mathbf{x} = \mathbf{b}_d$ can be solved efficiently, see also Thm. 3.3.3.

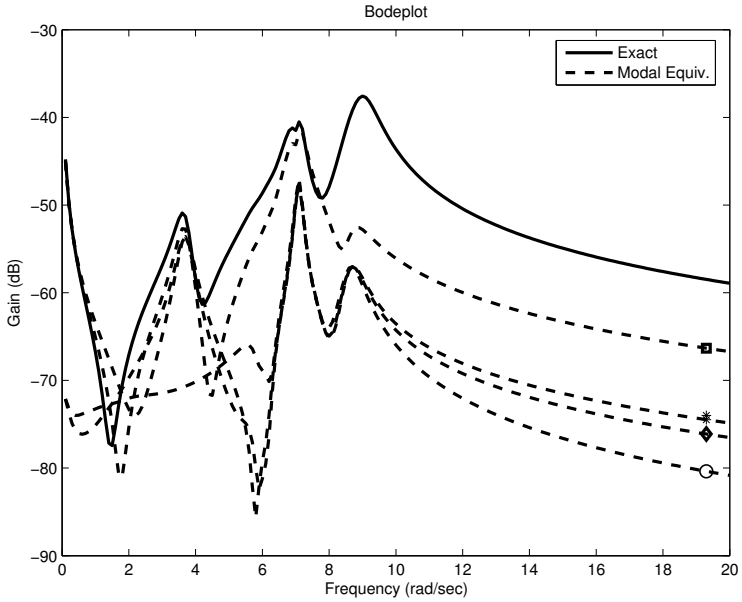

Figure 3.1: Exact transfer function (solid) of the New England test system [91], and modal equivalents where the following dominant pole (pairs) are removed one by one: $-0.467 \pm 8.96i$ (square), $-0.297 \pm 6.96i$ (asterisk), -0.0649 (diamond), and $-0.249 \pm 3.69i$ (circle). Note that the corresponding peaks are removed from the Bode plot as well (to see this, check the Bode plot at the frequencies near the imaginary part of the removed pole).

$(s_k I - \Lambda)^{-1} Y^* \mathbf{b}_d = 0$ and $\mathbf{w}^* E X = \mathbf{c}_d^* (s_k E - A)^{-1} E X = \mathbf{c}_d^* X (s_k I - \Lambda) = 0$. Consequently, $(I - XY^*E)\mathbf{v} = \mathbf{v}$, and since $(I - EXY^*)^2 = (I - EXY^*)$, $\mathbf{v}$ also satisfies $(s_k E_d - A_d)\mathbf{v} = \mathbf{b}_d$, and, similarly, $\mathbf{w}$ also satisfies $(s_k E_d - A_d)^* \mathbf{w} = \mathbf{c}_d$. It follows that if $\tilde{s}_k = s_k$, then

$$\tilde{s}_{k+1} = \frac{\mathbf{w}^* A_d \mathbf{v}}{\mathbf{w}^* E_d \mathbf{v}} = \frac{\mathbf{w}^* A \mathbf{v}}{\mathbf{w}^* E \mathbf{v}} = s_{k+1}$$

for all $k \geq 0$. □

The important consequence is that the single pole DPA can easily be extended, see Algorithm 3.1, to an algorithm that is able to compute more than one pole, while maintaining constant costs per iteration, except for iterations in which a pole is found. The only change to be made to Algorithm 2.1, is when a dominant pole triplet $(\lambda, \mathbf{x}, \mathbf{y})$ is found: in that case, the algorithm continues with $\mathbf{b}$ and $\mathbf{c}$ replaced by $(I - E\mathbf{x}\mathbf{y}^*)\mathbf{b}$ and $(I - E^*\mathbf{y}\mathbf{x}^*)\mathbf{c}$, respectively.

Theoretically speaking this approach has a number of advantages. The implementation is very straightforward and efficient: search spaces, selection strategies and orthogonalization procedures are not needed, so that the computational costs

per iteration remain constant, even if the number of found poles increases. For every found pole only two skew projections are needed once to compute the new $\mathbf{b}_d$ and $\mathbf{c}_d$, so the costs for deflation are constant. The pseudo code in Algorithm 3.1 can almost literally be used as Matlab code. The special properties of DPA ensure convergence to dominant poles (locally). Furthermore, the deflation of found poles is numerically stable in the sense that even if the corresponding transformed residues are not exactly zero, which is usually the case in finite arithmetic, this will hardly influence the effect of deflation: firstly, all the poles are left unchanged, and secondly, already a decrease of dominance of the found poles to nondominance (because of the projected in- and output vectors $\mathbf{b}_d$ and $\mathbf{c}_d$) will shrink the local convergence neighborhood of these poles significantly, again because of the convergence behavior of DPA [126] (see Chapter 2).

Algorithm 3.1 Dominant Pole Algorithm with deflation (DPAd)

INPUT: System $(E, A, \mathbf{b}, \mathbf{c})$, initial pole estimates $s_0^1, \ldots, s_0^p$, tolerance $\epsilon \ll 1$
OUTPUT: Approximate dominant poles $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_p)$, and corresponding right and left eigenvectors $X = [\mathbf{x}_1, \ldots, \mathbf{x}_p]$ and $Y = [\mathbf{y}_1, \ldots, \mathbf{y}_p]$

Set $k = 0$, $i = 0$, $s_k = s_0^1$
while $i < p$ **do**
  {Continue until p poles have been found}
  Solve $\mathbf{v}_k \in \mathbb{C}^n$ from $(s_k E - A)\mathbf{v}_k = \mathbf{b}$
  Solve $\mathbf{w}_k \in \mathbb{C}^n$ from $(s_k E - A)^* \mathbf{w}_k = \mathbf{c}$
  Compute the new pole estimate

$$s_{k+1} = s_k - \frac{\mathbf{c}^* \mathbf{v}_k}{\mathbf{w}_k^* E \mathbf{v}_k} = \frac{\mathbf{w}_k^* A \mathbf{v}_k}{\mathbf{w}_k^* E \mathbf{v}_k}$$

  if $\|A\mathbf{v}_k - s_{k+1} E \mathbf{v}_k\|_2 < \epsilon$ (with $\|\mathbf{v}_k\|_2 = 1$) **then**
    Set $i = i + 1$
    Set $\lambda_{ii} = s_{k+1}$
    Set $\mathbf{v}_k = \mathbf{v}_k / (\mathbf{w}_k^* E \mathbf{v}_k)$
    Set $X = [X, \mathbf{v}_k]$ and $Y = [Y, \mathbf{w}_k]$
    Deflate: $\mathbf{b} = \mathbf{b} - E\mathbf{v}_k \mathbf{w}_k^* \mathbf{b}$
    Deflate: $\mathbf{c} = \mathbf{c} - E^* \mathbf{w}_k \mathbf{v}_k^* \mathbf{c}$
    Set $s_{k+1} = s_0^i$
  end if
  Set $k = k + 1$
end while

This approach, however, may also suffer from several issues. Firstly, the convergence behavior can be very local and hence may heavily depend on the initial estimates s_0^i. Although in practice one often has rather accurate initial estimates of the poles of interest, this may be problematic if accurate information is not available. It may take many iterations until convergence if the initial estimate is not in the neighborhood of a dominant pole. On the other hand, the computational

complexity of this problem depends on the costs of the LU factorization, which in certain practical examples can be computed very efficiently.

Secondly, since the algorithm is usually applied to sparse descriptor systems, there are also eigenvalues of (A, E) at infinity. Although these eigenvalues are usually not dominant poles, since the corresponding residues are equal to zero, the estimates s_k computed by DPA may be approximations of the pole at infinity: due to rounding errors, the approximations $\mathbf{v}_k$ and $\mathbf{w}_k$ may be spoiled by components in the direction of eigenvectors corresponding to the eigenvalues at infinity. This may be prevented by purification techniques (see [97] and Chapter 7), but this does not fit nicely in the Newton scheme and costs (at least) two additional LU solves per iteration. If the poles at infinity are not dominant, DPA will never converge to such a pole, but convergence to other poles may be hampered nonetheless. Also, the presence of Jordan blocks may cause s_k to go to infinity: if λ has geometric multiplicity smaller than its algebraic multiplicity, then $\mathbf{y}^* E \mathbf{x} = 0$. If s_k happens to be a good estimate of λ, then $\mathbf{w}_k E \mathbf{v}_k \to 0$ and s_{k+1} can become very large (although two-sided Rayleigh quotient iteration may still converge at asymptotically linear rate, cf. [110, remark 2, p.689]). Problems due to the presence of Jordan blocks were, however, not experienced in any practical situation.

Thirdly, sooner or later the algorithm may get trapped in a stable periodic point of period 2, or even higher, of the underlying Newton algorithm. Such a point is in general not a zero (i.e. eigenvalue) of the original problem, but a periodic (fixed) point of the Newton scheme (which is a dynamical system itself). The chance that this happens increases as more poles are found. The difficulty here is that it is in general not possibly to predict the location or even the existence of stable periodic points a priori. The resulting stagnation, i.e. cycling between two complex conjugate values in the case of a period 2 point, however, can easily be detected by inspecting the values of s_k and s_{k+1}. Restarting with a different initial shift then solves the problem temporarily and may be sufficient in practice. In numerical experiments, periodic points were only observed after a large number of dominant poles had been found. To be more precise, for the cases where periodic points were encountered already all the dominant poles had been found. Intuitively one could argue that the remaining transfer function is of low magnitude (or even noise) and hence the chance of getting trapped in stable periodic points is bigger.

The main problem encountered in numerical experiments is slow global convergence. Combined with the characteristic convergence to dominant poles close to the initial shift, this stresses why the initial shifts are of great importance. If none of the shifts is in the neighborhood of the dominant pole, it may even not be found, due to the local character of the Newton process. On the other hand, provided the linear solves or LUs can be done cheaply, the low costs per iteration allow for the computation of poles for many different initial shifts, so that the frequency range of interest can be scanned effectively. In the next section a subspace accelerated version of DPA is described, that improves the global convergence by using search spaces.

3.3.2 Subspace accelerated DPA

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 current right and left approximate eigenvectors $\mathbf{v}_k$ and $\mathbf{w}_k$, however, may also contain components in the direction of eigenvectors corresponding to other dominant poles. Instead of discarding these approximate eigenvectors, they are kept in search spaces spanned by the columns of V and W, respectively. This idea is known as subspace acceleration.

A global overview of SADPA is shown in Algorithm 3.2. Starting with a single shift s_1, the first iteration is equivalent to the first iteration of the DPA (step 3-4). The right and left eigenvector approximations $\mathbf{v}_1$ and $\mathbf{w}_1$ are kept in spaces V and W. In the next iteration, these spaces are expanded orthogonally, by using modified Gram-Schmidt (MGS) [64] (see also Algorithm 1.3 in Section 1.5.4), with the approximations $\mathbf{v}_2$ and $\mathbf{w}_2$ corresponding to the new shift s_2 (step 5-6). Hence the spaces grow and will contain better approximations. Note that it is also possible to keep V and W bi-orthogonal, or bi-E-orthogonal, see Section 3.4.

Algorithm 3.2 Subspace Accelerated DPA (SADPA)

INPUT: System $(E, A, \mathbf{b}, \mathbf{c})$, initial pole estimate s_1 and the number of wanted poles p

OUTPUT: Dominant pole triplets $(\lambda_i, \mathbf{x}_i, \mathbf{y}_i)$, $i = 1, \ldots, p$

1: $k = 1$, $p_{found} = 0$, $\Lambda = [\,]$, $X = Y = [\,]$
2: **while** $p_{found} < p$ **do**
3: Solve $\mathbf{v}$ from $(s_k E - A)\mathbf{v} = \mathbf{b}$
4: Solve $\mathbf{w}$ from $(s_k E - A)^* \mathbf{w} = \mathbf{c}$
5: $\mathbf{v} = \mathrm{MGS}(V, \mathbf{v})$, $V = [V, \mathbf{v}/||\mathbf{v}||_2]$
6: $\mathbf{w} = \mathrm{MGS}(W, \mathbf{w})$, $W = [W, \mathbf{w}/||\mathbf{w}||_2]$
7: Compute $S = W^*AV$ and $T = W^*EV$
8: $(\tilde{\Lambda}, \tilde{X}, \tilde{Y}) = \mathrm{Sort}(S, T, W^*\mathbf{b}, V^*\mathbf{c})$ {Algorithm 3.3}
9: Dominant approximate eigentriplet of (A, E) is

$$(\hat{\lambda}_1 = \tilde{\lambda}_1, \hat{\mathbf{x}}_1 = V\tilde{\mathbf{x}}_1/\|V\tilde{\mathbf{x}}_1\|_2, \hat{\mathbf{y}}_1 = W\tilde{\mathbf{y}}_1/\|W\tilde{\mathbf{y}}_1\|_2)$$

10: **if** $||A\hat{\mathbf{x}}_1 - \hat{\lambda}_1 E\hat{\mathbf{x}}_1||_2 < \epsilon$ **then**
11: $(\Lambda, X, Y, V, W, \mathbf{b}, \mathbf{c}) =$
$\mathrm{Deflate}(\hat{\lambda}_1, \hat{\mathbf{x}}_1, \hat{\mathbf{y}}_1, \Lambda, X, Y, V\widetilde{X}_{2:k}, W\widetilde{Y}_{2:k}, E, \mathbf{b}, \mathbf{c})$ {Algorithm 3.4}
12: $p_{found} = p_{found} + 1$
13: Set $\tilde{\lambda}_1 = \tilde{\lambda}_2$, $k = k - 1$
14: **end if**
15: Set $k = k + 1$
16: Set the new pole estimate $s_k = \tilde{\lambda}_1$
17: **end while**

Selection strategy

In iteration k the Petrov-Galerking approach leads (step 7) to the projected eigenproblem

$$\begin{aligned} W^*AV\tilde{\mathbf{x}} &= \tilde{\lambda}W^*EV\tilde{\mathbf{x}}, \\ \tilde{\mathbf{y}}W^*AV &= \tilde{\lambda}\tilde{\mathbf{y}}W^*EV. \end{aligned}$$

Since the interaction matrices $S = W^*AV$ and $T = W^*EV$ are of low dimension $k \ll n$, the eigentriplets $(\tilde{\lambda}_i, \tilde{\mathbf{x}}_i, \tilde{\mathbf{y}}_i)$ of this reduced problem can be computed using the QZ method (or the QR method in the bi-E-orthogonal case (step 1 of Algorithm 3.3)). This provides k approximate eigentriplets $(\hat{\lambda}_i = \tilde{\lambda}_i, \hat{\mathbf{x}}_i = V\tilde{\mathbf{x}}_i, \hat{\mathbf{y}}_i = W\tilde{\mathbf{y}}_i)$ for (A, E). The most natural thing to do is to choose the triplet $(\hat{\lambda}_j, \hat{\mathbf{x}}_j, \hat{\mathbf{y}}_j)$ with the most dominant pole approximation (step 8-9): compute the corresponding residues $\widehat{R}_i = (\mathbf{c}^*\hat{\mathbf{x}}_i)(\hat{\mathbf{y}}_i^*\mathbf{b})$ of the k pairs and select the pole with the largest $|\widehat{R}_j|/|\mathrm{Re}(\hat{\lambda}_j)|$ (see Algorithm 3.3). The SADPA then continues with the new shift $s_{k+1} = \hat{\lambda}_j$ (step 16).

The residues $\widehat{R}_i$ can be computed without computing the approximate eigenvectors explicitly (step 5 of Algorithm 3.3): if the $\tilde{\mathbf{x}}_i$ and $\tilde{\mathbf{y}}_i$ are scaled so that $\tilde{\mathbf{y}}_i^*T\tilde{\mathbf{x}}_i = 1$ $(= \hat{\mathbf{y}}_i^*E\hat{\mathbf{x}}_i)$, then it follows that the $\widehat{R}_i$ can be computed as $\widehat{R}_i = ((\mathbf{c}^*V)\tilde{\mathbf{x}}_i)(\tilde{\mathbf{y}}_i^*(W^*\mathbf{b}))$ $(= (\mathbf{c}^*\hat{\mathbf{x}}_i)(\hat{\mathbf{y}}_i^*\mathbf{b}))$.

Instead of $\hat{\mathbf{y}}_i^*E\hat{\mathbf{x}}_i = 1$ one can also use the scaling $\|\hat{\mathbf{y}}_i\|_2 = \|\hat{\mathbf{x}}_i\|_2 = 1$ when computing approximate residues. In that case the product of the angles $\angle(\hat{\mathbf{x}}_i, \mathbf{c})$ and $\angle(\hat{\mathbf{y}}_i, \mathbf{b})$ is used in the computation of the approximate residues (see also Chapter 2), which numerically may be more robust. See Section 3.7 and the Addendum of Chapter 4 for more details.

Algorithm 3.3 $(\widetilde{\Lambda}, \widetilde{X}, \widetilde{Y}) = \mathrm{Sort}(S, T, \mathbf{b}, \mathbf{c})$

INPUT: $S, T \in \mathbb{C}^{k\times k}$, $\mathbf{b}, \mathbf{c} \in \mathbb{C}^k$

OUTPUT: $\widetilde{\Lambda} \in \mathbb{C}^k$, $\widetilde{X}, \widetilde{Y} \in \mathbb{C}^{k\times k}$ with λ_1 the pole with largest (scaled) residue magnitude and $\tilde{\mathbf{y}}_1$ and $\tilde{\mathbf{x}}_1$ the corresponding right and left eigenvectors.

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

$$(\tilde{\lambda}_i, \tilde{\mathbf{x}}_i, \tilde{\mathbf{y}}_i), \quad \tilde{\mathbf{y}}_i^*T\tilde{\mathbf{x}}_i = 1, \quad i = 1, \ldots, k$$

2: $\widetilde{\Lambda} = [\tilde{\lambda}_1, \ldots, \tilde{\lambda}_k]$
3: $\widetilde{X} = [\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_k]$
4: $\widetilde{Y} = [\tilde{\mathbf{y}}_1, \ldots, \tilde{\mathbf{y}}_k]$
5: Compute residues $R_i = (\mathbf{c}^*\tilde{\mathbf{x}}_i)(\tilde{\mathbf{y}}_i^*\mathbf{b})$
6: Sort $\widetilde{\Lambda}$, $\widetilde{X}$, $\widetilde{Y}$ in decreasing $|R_i|/|\mathrm{Re}(\tilde{\lambda}_i)|$ order

---

Deflation

Every iteration a convergence test (step 10) is done like in DPAd (Algorithm 3.1): if for the selected eigentriplet $(\hat{\lambda}_1, \hat{\mathbf{x}}_1, \hat{\mathbf{y}}_1)$ the norm of the residual $||A\hat{\mathbf{x}}_1 - \hat{\lambda}_1\hat{\mathbf{x}}_1||_2$ is smaller than some tolerance ϵ, it is converged. In general more than one dominant eigentriplet is wanted and it is desirable to avoid repeated computation of the same eigentriplet. The same deflation technique as used in DPAd can be applied here (step 5-6 and 12-13 of Algorithm 3.4, see also Section 3.3.1), and since SADPA continues with $\mathbf{b}_d$ and $\mathbf{c}_d$, no explicit E-orthogonalization of expansion vectors against found eigenvectors is needed in step 3 and 4. The effect is similar to the usual deflation in Jacobi-Davidson methods (see [51] and also Section 3.6.2): found eigenvectors are hard-locked, i.e. once deflated, they do not participate and do not improve during the rest of the process (contrary to soft-locking, where deflated eigenvectors still participate in the Rayleigh-Ritz (Ritz-Galerkin) procedure and may be improved, at the cost of additional computations and administration, see [80, 81]). In fact, there is cheap explicit deflation without the need for implicit deflation (cf. [51, remark 5, p. 106], where a combination of explicit and implicit deflation is used).

Algorithm 3.4 $(\Lambda, X, Y, \widetilde{V}, \widetilde{W}, \mathbf{b}_d, \mathbf{c}_d) = \text{Deflate}(\lambda, \mathbf{x}, \mathbf{y}, \Lambda, X, Y, V, W, E, \mathbf{b}, \mathbf{c})$

INPUT: $\lambda \in \mathbb{C}$, $\mathbf{x}, \mathbf{y} \in \mathbb{C}^n$, $\Lambda \in \mathbb{C}^p$, $X, Y \in \mathbb{C}^{n\times p}$, $V, W \in \mathbb{C}^{n\times k}$, $E \in \mathbb{C}^{n\times n}$, $\mathbf{b}, \mathbf{c} \in \mathbb{C}^n$

OUTPUT: $\Lambda \in \mathbb{C}^q$, $X, Y \in \mathbb{C}^{n\times q}$,$\widetilde{V}, \widetilde{W} \in \mathbb{C}^{n\times k-1}$,$\mathbf{b}_d, \mathbf{c}_d \in \mathbb{C}^n$, 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: Set $\mathbf{x} = \mathbf{x}/(\mathbf{y}^* E\mathbf{x})$
3: $X = [X, \mathbf{x}]$
4: $Y = [Y, \mathbf{y}]$
5: Deflate: $\mathbf{b}_d = \mathbf{b} - E\mathbf{x}(\mathbf{y}^*\mathbf{b})$
6: Deflate: $\mathbf{c}_d = \mathbf{c} - E^*\mathbf{y}(\mathbf{x}^*\mathbf{c})$
7: **if** imag$(\lambda) \neq 0$ **then**
8: {Also deflate complex conjugate}
9: $\Lambda = [\Lambda, \bar{\lambda}]$
10: $\mathbf{x} = \bar{\mathbf{x}}$, $X = [X, \mathbf{x}]$
11: $\mathbf{y} = \bar{\mathbf{y}}$, $Y = [Y, \mathbf{y}]$
12: Deflate: $\mathbf{b}_d = \mathbf{b}_d - E\mathbf{x}(\mathbf{y}^*\mathbf{b}_d)$
13: Deflate: $\mathbf{c}_d = \mathbf{c}_d - E^*\mathbf{y}(\mathbf{x}^*\mathbf{c}_d)$
14: **end if**
15: $\widetilde{V} = \widetilde{W} = [\ ]$
16: **for** $j = 1, \ldots, k$ **do**
17: $\widetilde{V} = \text{Expand}(\widetilde{V}, X, Y, E, \mathbf{v}_j)$ {Algorithm 3.5}
18: $\widetilde{W} = \text{Expand}(\widetilde{W}, Y, X, E^*, \mathbf{w}_j)$ {Algorithm 3.5}
19: **end for**

If an eigentriplet has converged (step 11-13), the eigenvectors are deflated from

the search spaces by reorthogonalizing the search spaces against the found eigenvectors. This can be done by using modified Gram-Schmidt (MGS) and by recalling that, if the exact vectors are found, the pencil

$$((I - EXY^*)A(I - XY^*E), \quad (I - EXY^*)E(I - XY^*E))$$

has the same eigentriplets as (A, E), but with the found eigenvalues transformed to zero (Algorithm 3.5, see also [51, 74]). Since in finite arithmetic only *approximations* to exact eigentriplets are available, the computed eigenvalues are transformed to $\eta \approx 0$. The possible numerical consequences of this, however, are limited, since SADPA continues with $\mathbf{b}_d$ and $\mathbf{c}_d$, and as argued in Section 3.3.1, the residues of the found poles are transformed to (approximately) zero.

If a complex pole has converged, its complex conjugate is also a pole and the corresponding complex conjugate right and left eigenvectors can also be deflated. A complex conjugate pair is counted as one pole. The complete deflation procedure is shown in Algorithm 3.4.

After deflation of the found pole(s), SADPA continues with the second most dominant approximate pole (step 13-16).

Algorithm 3.5 $V = \text{Expand}(V, X, Y, E, \mathbf{v})$

INPUT: $V \in \mathbb{C}^{n\times k}$ with $V^*V = I$, $X, Y \in \mathbb{C}^{n\times p}$, $E \in \mathbb{C}^{n\times n}$, $\mathbf{v} \in \mathbb{C}^n$, Y^*EX diagonal, $Y^*EV = 0$

OUTPUT: $V \in \mathbb{C}^{n\times(k+1)}$ with $V^*V = I$ and $\mathbf{v}_{k+1} = \prod_{j=1}^{p}(I - \frac{\mathbf{x}_j\mathbf{y}_j^*E}{\mathbf{y}_j^*E\mathbf{x}_j}) \cdot \mathbf{v}$

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

Further improvements and remarks

It may happen that the search spaces V and W become high-dimensional, especially when a large number of dominant poles is wanted. A common way to deal with this is to do an thick restart [51, 132]: if the subspaces V and W 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 V and W (already converged eigentriplets are not part of the active subspaces V and W). This procedure is repeated until all poles are found.

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 $\epsilon \leq 10^{-10}$. Besides that, as the estimate converges to a dominant pole, the right and left eigenvectors computed in step 3 and 4 of Algorithm 3.2 are usually more accurate than the approximations

computed in the selection procedure: if the estimate s_k is close to an eigenvalue λ, then $s_kE - A$ may become ill conditioned, but, as is discussed in [116] and [112, Section 4.3], the solutions $\mathbf{v}_k$ and $\mathbf{w}_k$ have large components in the direction of the corresponding right and left eigenvectors (provided $\mathbf{b}$ and $\mathbf{c}$ have sufficiently large components in those directions). In the deflation phase, it is therefore advised to take the most accurate of both, i.e., the approximate eigenvector with smallest residual. It may also be advantageous to do an additional step of two-sided Rayleigh quotient iteration to improve the eigenvectors (see Section 3.5.2 for more details).

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 16 of Algorithm 3.2).

Every iteration, two linear systems are to be solved (step 3 and 4). As was also mentioned in Section 3.2.1, this can efficiently be 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 matrices A and E are often very sparse and structured, computation of the LU-factorizations can be relatively inexpensive.

Although poles at infinity are usually not dominant, components in the direction of the corresponding eigenvectors may still enter the search spaces (as was observed occasionally). Most likely these components are removed from the search spaces at a restart, but they might enter the search spaces again during the remainder of the process. A way to prevent this is to keep these components in the search spaces at a restart. The selection criterion takes care of selection of the approximations to finite dominant poles. In Chapter 7 more sophisticated purification techniques are studied.

The selection criterion can easily be changed to another of the several existing indices of modal dominance [4, 68, 159]. 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 evaluated efficiently for the frequency range of interest.

A numerical example

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 [91]). The tolerance used was $\epsilon = 10^{-10}$ and no restarts were used. Every iteration, the pole approximation $\hat{\lambda}_j$ with largest $|\widehat{R}_j|/|\mathrm{Re}(\hat{\lambda}_j)|$ was selected. Table 3.1 shows the found dominant poles and the iteration number for which the pole satisfied the stopping criterion. Bodeplots of two modal equivalents are shown in Figure 3.2. 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. In Section 3.7,

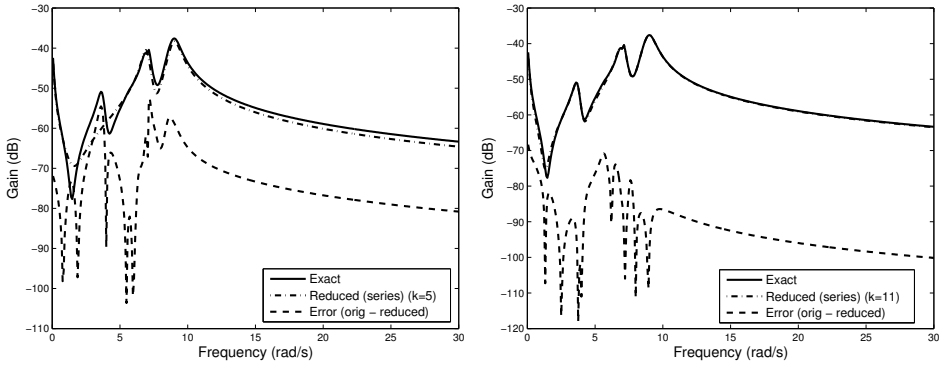

Figure 3.2: Bode plot of 5th order (left) and 11th order (right) modal equivalent, complete model and error for the transfer function of the New England test system (66 states in the complete model).

large-scale numerical examples are considered.

Table 3.1: 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	Figure 3.2 (left)
4	7	$-0.2491 \pm 3.6862i$	25	-
5	9	$-0.1118 \pm 7.0950i$	26	-
6	11	$-0.3704 \pm 8.6111i$	27	Figure 3.2 (right)

3.3.3 Comparison between DPAd and SADPA

The expectation is that the use of search spaces not only improves the global convergence, but also reduces the average number of iterations needed to converge to a pole. This was confirmed by numerical experiments. In Figure 3.3 the average number of iterations and time per pole are shown, as well as the average residue norm of the found poles, for SADPA and DPAd. The test system (BIPS, see Section 3.7 for more details) was a descriptor system of dimension $n = 13,251$. It is clear that SADPA is faster than DPAd and requires less LU-factorizations. Since SADPA selects the most dominant approximation using $|R_i/\mathrm{Re}(\lambda_i)|$ as index for dominance, it converges to more dominant poles (measured using this index). Measured in the index $|R_i|$, DPAd computes more dominant poles on the average for the first 60 poles. However, if SADPA uses this index as selection criterion, it outperforms DPAd also for smaller numbers of poles.

The additional costs for SADPA for keeping the search spaces are earned back by faster global convergence: as a pole has converged and its corresponding eigen-

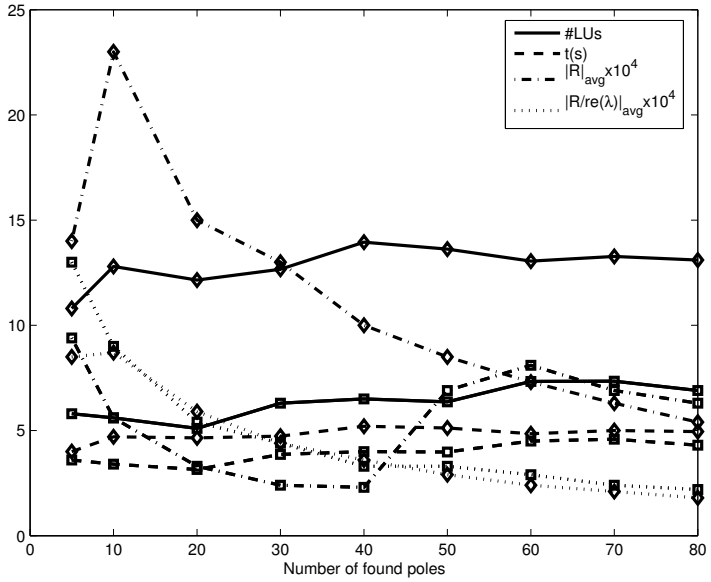

Figure 3.3: Average number of LUs (solid), time (dashed), residue norm $|R|$ (dash-dot) and scaled residue norm $|R|/|\mathrm{Re}(\lambda)|$ (dotted) per pole for SADPA (squares) and DPAd (diamonds). SADPA is faster, requires less LUs and has a larger average scaled residue norm than DPAd.

vectors are deflated from the search spaces, the remaining search spaces contain approximations of eigenvectors corresponding to other dominant poles. Hence, SADPA continues with a rather promising approximation of the next pole, while DPAd has to (re)start from scratch. The additional memory requirements are also limited since SADPA is restarted at dimension $k_{max} = 6$ to search spaces of dimension $k_{min} = 2$.

In practice, SADPA is the method of choice for many problems, because of its robustness, flexibility and fast convergence. For a wide range of problems, varying from small to large-scale systems, $1 \leq k_{min} < k_{\max} \leq 10$ are appropriate settings. The selection criterion can easily be adapted to the preferred measure of dominance. If, on the other hand, a small number of dominant poles in the close vicinity of some accurate initial shifts are wanted, and the LU factorizations can be computed efficiently, then it may be more advantageous to use DPAd, because of its local character.

3.4 SADPA and two-sided Jacobi-Davidson

3.4.1 Two-sided Jacobi-Davidson

If in the k-th iteration of SADPA, the right-hand sides $\mathbf{b}$ and $\mathbf{c}$ are replaced by $E\mathbf{v}_k$ and $E^*\mathbf{w}_k$, where $\mathbf{v}_k$ and $\mathbf{w}_k$ are the right and left approximate eigenvectors corresponding to the selected s_k, then the algorithm becomes subspace accelerated two-sided RQI. In this section, two-sided Jacobi-Davidson [74, 139, 148] is described. As will be shown in Section 3.4.2, all three methods are equivalent if the (correction) equations are solved exactly.

Two-sided Jacobi-Davidson (BiJD) [74, 148] is the Jacobi-Davidson [140] analog of two-sided Rayleigh quotient iteration [107, 110]. Let the columns of $V \in \mathbb{C}^{n\times k}$ and $W \in \mathbb{C}^{n\times k}$ be (orthogonal or bi-orthogonal) bases for the k-dimensional right and left search spaces $\mathcal{V}$ and $\mathcal{W}$. Two questions arise: (1) how can approximate eigentriplets be extracted from the search spaces, and (2) how can the search spaces be expanded in an effective way. To determine right and left approximate eigenvectors $\mathbf{v} = V\mathbf{s}$ and $\mathbf{w} = W\mathbf{t}$, with $\mathbf{s}, \mathbf{t} \in \mathbb{C}^k$, Petrov-Galerkin conditions are imposed on the right residual $\mathbf{r}_v$ and the left residual $\mathbf{r}_w$:

$$\mathbf{r}_v = A\mathbf{v} - \theta E\mathbf{v} \perp W \quad \text{and} \quad \mathbf{r}_w = A^*\mathbf{w} - \bar{\theta}E^*\mathbf{w} \perp V,$$

where $\theta = (\mathbf{w}^*A\mathbf{v})/(\mathbf{w}^*E\mathbf{v})$. It follows that $(\theta, \mathbf{s}, \mathbf{t})$ can be computed as an eigentriplet of the projected eigenproblem

$$W^*AV\mathbf{s} = \theta W^*EV\mathbf{s} \quad \text{and} \quad V^*A^*W\mathbf{t} = \bar{\theta}V^*E^*W\mathbf{t}.$$

Note that this is a problem of small size $k \ll n$ that can be solved using a full space method like the QZ method. When computing dominant poles, the selection strategy used in SADPA (Algorithm 3.3) can be used to select to most dominant approximate triplet.

Given an approximate eigentriplet $(\theta, \mathbf{v}, \mathbf{w})$, two-sided JD computes corrections $\mathbf{f}$ and $\mathbf{g}$ so that

$$A(\mathbf{v}+\mathbf{f}) = \lambda E(\mathbf{v}+\mathbf{f}) \quad \text{and} \quad A^*(\mathbf{w}+\mathbf{g}) = \bar{\lambda}E^*(\mathbf{w}+\mathbf{g}),$$

where λ is unknown. This leads to the right correction equation

$$(A-\lambda E)\mathbf{f} = -\mathbf{r}_v + (\lambda-\theta)\mathbf{v},$$

and the left correction equation

$$(A-\lambda E)^*\mathbf{g} = -\mathbf{r}_w + (\bar{\lambda}-\bar{\theta})\mathbf{w}.$$

Typical for the JD process, these equations are projected so that $E\mathbf{v}$ and $E^*\mathbf{w}$ are mapped to 0, respectively, and so that $\mathbf{r}_v$ and $\mathbf{r}_w$ are kept fixed. Replacing λ by θ, this gives the correction equations

$$\begin{aligned}\left(I - \frac{E\mathbf{v}\mathbf{w}^*}{\mathbf{w}^*E\mathbf{v}}\right)(A-\theta E)\mathbf{f} &= -\mathbf{r}_v,\\ \left(I - \frac{E^*\mathbf{w}\mathbf{v}^*}{\mathbf{v}^*E^*\mathbf{w}}\right)(A-\theta E)^*\mathbf{g} &= -\mathbf{r}_w.\end{aligned}$$

There are two obvious choices for the search spaces: either they are kept bi-E-orthogonal, i.e. $W^*EV = I$, or they are kept orthogonal, i.e. $W^*W = V^*V = I$. If V and W are kept bi-E-orthogonal, which seems natural since the right and left eigenvectors are also bi-E-orthogonal, the correction equations become

$$\left(I - \frac{E\mathbf{v}\mathbf{w}^*}{\mathbf{w}^*E\mathbf{v}}\right)(A - \theta E)\left(I - \frac{\mathbf{v}\mathbf{w}^*E}{\mathbf{w}^*E\mathbf{v}}\right)\mathbf{f} = -\mathbf{r}_v \quad (\mathbf{f} \perp E^*\mathbf{w}), \quad (3.4.1)$$

$$\left(I - \frac{E^*\mathbf{w}\mathbf{v}^*}{\mathbf{v}^*E^*\mathbf{w}}\right)(A - \theta E)^*\left(I - \frac{\mathbf{w}\mathbf{v}^*E^*}{\mathbf{v}^*E^*\mathbf{w}}\right)\mathbf{g} = -\mathbf{r}_w \quad (\mathbf{g} \perp E\mathbf{v}). \quad (3.4.2)$$

If V and W are kept orthogonal, the correction equations become

$$\left(I - \frac{E\mathbf{v}\mathbf{w}^*}{\mathbf{w}^*E\mathbf{v}}\right)(A - \theta E)\left(I - \frac{\mathbf{v}\mathbf{v}^*}{\mathbf{v}^*\mathbf{v}}\right)\mathbf{f} = -\mathbf{r}_v \quad (\mathbf{f} \perp \mathbf{v}), \quad (3.4.3)$$

$$\left(I - \frac{E^*\mathbf{w}\mathbf{v}^*}{\mathbf{v}^*E^*\mathbf{w}}\right)(A - \theta E)^*\left(I - \frac{\mathbf{w}\mathbf{w}^*}{\mathbf{w}^*\mathbf{w}}\right)\mathbf{g} = -\mathbf{r}_w \quad (\mathbf{g} \perp \mathbf{w}). \quad (3.4.4)$$

Keeping V and W bi-E-orthogonal has the advantage that the projected problem reduces to an ordinary eigenproblem, but the disadvantage that the E-inner products are more expensive than standard inner products and possibly less stable or even not well-defined. There are also other options for the projections in the correction equations, see [139]. See Alg. 1.5 for BiJD with orthogonal search spaces.

If the correction equations are solved exactly, then BiJD is equivalent to accelerated two-sided Rayleigh quotient iteration, see [74]. Note again that the projected operators are never formed explicitly. The solution for the right correction equation (3.4.1) with bi-E-orthogonal search spaces, for instance, can be computed as

$$\mathbf{f} = -\mathbf{v} + \mu(A - \theta E)^{-1}E\mathbf{v},$$

with $\mu = \mathbf{w}^*E\mathbf{v}/(\mathbf{w}^*E(A - \theta E)^{-1}E\mathbf{v})$. In practice it is often sufficient and even advantageous to solve the correction equation up to a certain precision with Krylov subspace methods. Inexact variants are discussed in Section 3.6.

3.4.2 Equivalence of SADPA and two-sided JD

While DPA and two-sided Rayleigh quotient iteration have different convergence behavior (locally quadratic versus cubic rate of convergence, and, moreover, DPA has better convergence to dominant poles, see also Chapter 2), their subspace accelerated versions can be shown to be equivalent if the linear systems are solved exactly (see also [74] for the equivalence of two-sided JD and two-sided RQI). This may be somewhat surprising, but with the following two lemmas it can be shown that the search spaces constructed by both algorithms are the same, provided the same initial vectors and selection criterion are used.

Lemma 3.4.1. *Let* $\sigma \neq \tau \in \mathbb{C} \setminus \Lambda(A, E)$, $\mathbf{x} = (\sigma E - A)^{-1}\mathbf{b}$, $\mathbf{y} = (\tau E - A)^{-1}\mathbf{b}$ *and* $\mathbf{z} = (\tau E - A)^{-1}E(\sigma E - A)^{-1}\mathbf{b}$. *Then* $\mathbf{z} \in$ *span*$(\mathbf{x}, \mathbf{y})$.

Proof. Suppose $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ are independent. Then for α, β, γ,

$$\alpha\mathbf{x} + \beta\mathbf{y} + \gamma\mathbf{z} = 0 \Longleftrightarrow \alpha = \beta = \gamma = 0.$$

Multiplication by $(\tau E - A) = (\sigma E - A + (\tau - \sigma)E)$ gives

$$\alpha\mathbf{b} + \alpha(\tau - \sigma)E(\sigma E - A)^{-1}\mathbf{b} + \beta\mathbf{b} + \gamma E(\sigma E - A)^{-1}\mathbf{b} = 0,$$

and the contradiction follows for $\alpha = -\gamma/(\tau - \sigma)$ and $\beta = -\alpha$. □

This result can be generalized in the following way.

Lemma 3.4.2. *Let $\sigma_i \in \mathbb{C} \setminus \Lambda(A, E)$ for $i = 0, 1, \ldots, k$, and let*

$$V_k = \mathit{span}((\sigma_0 E - A)^{-1}\mathbf{b}, (\sigma_1 E - A)^{-1}\mathbf{b}, \ldots, (\sigma_{k-1} E - A)^{-1}\mathbf{b}).$$

If $\mathbf{v}_k \in V_k$, then $\mathbf{v}_{k+1} = (\sigma_k E - A)^{-1} E\mathbf{v}_k \in V_{k+1}$.

Proof. Since $\mathbf{v}_k \in V_k$, there is a $\mathbf{u} \in \mathbb{C}^k$ so that $\mathbf{v}_k = V_k\mathbf{u}$. By lemma 3.4.1 it follows that

$$(\sigma_k E - A)^{-1} E(\sigma_i E - A)^{-1}\mathbf{b} \in \mathrm{span}((\sigma_i E - A)^{-1}\mathbf{b}, (\sigma_k E - A)^{-1}\mathbf{b})$$

for $i = 0, 1, \ldots, k-1$. Hence, $\mathbf{v}_{k+1} \in V_{k+1}$. □

In other words, given a search space V_k and a Ritz pair $(s_k, \mathbf{v})$ with $\mathbf{v} \in V_k$, the search space V_k expanded with the new approximation $\mathbf{v}_{k+1} = (s_k E - A)^{-1} E\mathbf{v}$ (RQI) is equal to V_k expanded with $\mathbf{v}_{k+1} = (s_k E - A)^{-1}\mathbf{b}$ (DPA). A similar conclusion can be drawn for the left search space W_k with expansions $\mathbf{w}_{k+1} = (s_k E - A)^{-*} E^*\mathbf{w}$ or $\mathbf{w}_{k+1} = (s_k E - A)^{-*}\mathbf{c}$. Consequently, accelerated DPA and accelerated two-sided RQI are equivalent, provided that if DPA is started with s_0, then two-sided RQI has to be started with $\mathbf{v}_0 = (s_0 E - A)^{-1}\mathbf{b}$ and $\mathbf{w}_0 = (s_0 E - A)^{-*}\mathbf{c}$, and both methods use the same selection criterion.

Theorem 3.4.3. *Let $s_0 \in \mathbb{C}$, $\mathbf{v}_0 = (s_0 E - A)^{-1}\mathbf{b}$ and $\mathbf{w}_0 = (s_0 E - A)^{-*}\mathbf{c}$. If the same selection criterion is used, then SADPA($E, A, \mathbf{b}, \mathbf{c}, s_0$), exact two-sided RQI($E, A, \mathbf{v}_0, \mathbf{w}_0$), and exact BiJD($E, A, \mathbf{v}_0, \mathbf{w}_0$) are equivalent.*

Proof. Use lemma 3.4.2 and an induction argument. For the equivalence of BiJD and two-sided RQI, see [74]. □

Note that it is not only crucial that two-sided RQI starts with the specific $\mathbf{v}_0 = (s_0 E - A)^{-1}\mathbf{b}$ and $\mathbf{w}_0 = (s_0 E - A)^{-*}\mathbf{c}$, but that this is also a natural choice: given an initial shift s_0 it is (locally) the best choice, and, secondly, in general $E\mathbf{b} = 0$ and/or $E^*\mathbf{c} = 0$, so that the choices $\mathbf{v}_0 = (s_0 E - A)^{-1} E\mathbf{b}$ and $\mathbf{w}_0 = (s_0 E - A)^{-*} E^*\mathbf{c}$ are not applicable at all. Strictly speaking, BiJD (and subspace accelerated Rayleigh quotient iteration) start with search spaces of dimension 1, while SADPA starts with empty search spaces.

The advantage of SADPA over BiJD and two-sided RQI is that SADPA can make use of very cheap deflation via $\mathbf{b}_d = (I - E\mathbf{x}\mathbf{y}^*)\mathbf{b}$ and $\mathbf{c}_d = (I - E^*\mathbf{y}\mathbf{x}^*)\mathbf{c}$. Since for BiJD and two-sided RQI the relation with $\mathbf{b}$ and $\mathbf{c}$ is lost (except for the initial vectors), vectors that are added to the search spaces need to be orthogonalized against found eigenvectors. This needs to be done every iteration and becomes more expensive as more eigenvectors are found, while SADPA only has to deflate the found eigenvectors once during the whole process. To conclude, if it is affordable to solve the (correction) equations exactly, then SADPA is the method of choice. Numerical experiments confirm the equivalence of SADPA, BiJD and two-sided RQI, but also show that numerical round off may spoil the processes (see the example in the Section 3.5.2). In Section 3.6 it is shown that inexact BiJD is to be preferred if exact solves of the (correction) equations are not affordable.

The right and left search spaces constructed by SADPA (and accelerated two-sided RQI) are also known as rational Krylov subspaces [131]. In [131, Section 6] it is shown that for certain choices of the shifts and Ritz pairs, the rational Krylov scheme (RKS) is equivalent to the Jacobi-Davidson method. Generalizing, it can be shown in a similar way that SADPA and two-sided RKS are equivalent. See [131] and references therein for more details about the rational Krylov scheme, and [58] for the rational Lanczos process in the context of model order reduction.

3.5 The role of subspace acceleration

If a subspace accelerated process such as SADPA or BiJD is not restarted when a certain maximum dimension k_{max} of the search space is reached, then (in exact arithmetic) it trivially terminates within n iterations. In practice, usually good approximations of the wanted eigentriplet can be extracted from search spaces of dimension $k \ll n$. Together with restarting this makes the accelerated schemes preferable over the standard DPA and two-sided RQI, that are not guaranteed to converge within n iterations, or even do not converge at all in some situations (although this may also not be guaranteed when using restarts). For inexact versions the effects of subspace acceleration may have even bigger impact. Other effects of subspace acceleration are discussed in the following subsections.

3.5.1 Global convergence

Subspace acceleration not only speeds up the convergence, it also improves global convergence. Firstly, in the case of SADPA, it tempers the local character of DPA: since DPA is a Newton scheme, it converges most likely to a pole in the neighborhood of the initial guess. The search spaces may contain approximations of more dominant eigentriplets, that can be extracted using a selection criterion. Note that the use of subspace acceleration also offers flexibility with respect to the selection criterion, and provides some control on the convergence region.

Secondly, the search spaces automatically provide approximations for other dominant eigentriplets, so that after deflation of a found eigentriplet, the search for

the next can be continued with the most promising or dominant approximation. Also, if the initial guess is not close to a dominant pole, search spaces may quickly provide a better approximation. Note that in the case of DPAd a new shift should be supplied by the user to obtain better global convergence (although the same initial shift could be used after deflation without the risk of converging to the same pole).

As is also described in Section 3.3.2, the costs for keeping the (bi)-orthogonal search spaces is earned back by the faster convergence, lower overall computational time and qualitatively more dominant poles.

3.5.2 Local convergence

The situation is different in the final (local) phase of convergence to a pole. Suppose that the current approximation $(\hat{\lambda}, \mathbf{v}, \mathbf{w})$ of $(\lambda, \mathbf{x}, \mathbf{y})$ satisfies $||\mathbf{r}_v||_2 = ||A\mathbf{v} - \hat{\lambda}E\mathbf{v}||_2 = \tau$, with $\epsilon = 10^{-10} < \tau < 10^{-4}$. If the (correction) equations are solved exactly, then from now on it is not likely that much new information will be added to the search spaces, since $(\hat{\lambda}, \mathbf{v}, \mathbf{w}) \rightarrow (\lambda, \mathbf{x}, \mathbf{y})$ (up to $||\mathbf{r}_v||_2 < \epsilon$) within a few iterations. Instead of (bi)orthogonally expanding the search spaces, it is more efficient to improve the approximate eigentriplet using DPA or two-sided RQI. Usually this only requires 1 or 2 iterations, since the initial vectors are very close to the wanted eigenvectors and $\hat{\lambda} \approx \lambda$. For the same reason, there is low risk to converge to a pole $\mu \neq \lambda$, so deflation of already found eigentriplets is not needed in the DPA and RQI iterations.

Another problem that might occur is that round off errors, introduced by deflation, orthogonalization, and extraction of the eigentriplets from the projected problem, prevent the approximate eigentriplet $(\hat{\lambda}, \mathbf{v}, \mathbf{w})$ from satisfying $||\mathbf{r}_v||_2 = ||A\mathbf{v} - \hat{\lambda}E\mathbf{v}||_2 < \epsilon$: there is stagnation at $||\mathbf{r}_v|| = \tau$ with $\epsilon \lesssim \tau$. Also in this case two-sided RQI may help, since round off due to the inversion of $(A - \hat{\lambda}E)$ is in the desired direction [116], [112, Section 4.3] and the process is not disturbed by orthogonalization and deflation.

In practice the tolerance $\tau \ll 1$ for switching to DPA or two-sided RQI should be chosen with care. If τ is too large, the process of finding the most dominant will be more or less randomized and become greedy, resulting in less dominant poles. On the other hand, if τ is too small, the process may be hampered by stagnation or slow convergence. Based on experimental experience, a value $10^{-4} < \tau < 10^{-7}$ is a good choice in practice.

Numerical experiments show that a switch to two-sided RQI is more crucial for SADPA than for BiJD. The example displayed in Figure 3.4 is typical for this phenomenon. The left figure shows the convergence for SADPA and BiJD with tolerance $\epsilon = 10^{-8}$ for computing the 5 most dominant poles of the PEEC example from [28], *without* switching. Up to the 14th iteration the equivalence of both methods is apparent. However, SADPA, without switching, stagnates after iteration 17, while BiJD converges smoothly. This can be explained by the fact that Jacobi-Davidson computes the expansion vectors as corrections to the cur-

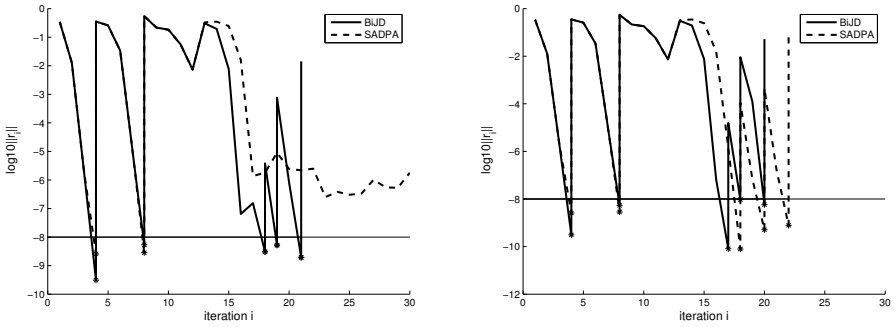

Figure 3.4: Convergence histories for BiJD and SADPA without switch (left) and with switch to two-sided RQI at $||\mathbf{r}_v||_2 = 10^{-6}$. The system is the PEEC example from [28], of order $n = 480$.

rent approximations, while SADPA in the final phase computes expansion vectors that make a (very) small angle with the current search spaces, possibly leading to cancellation during the orthogonal expansion. Also, the fixed right-hand sides in SADPA might hamper convergence to the desired tolerance.

The right history shows the convergence of SADPA and BiJD with switching to two-sided RQI at $||\mathbf{r}_v||_2 = 10^{-6}$. It can be observed that both methods take advantage of the switch to two-sided RQI, both with respect to the number of iterations and the accuracy. Note that SADPA now converges almost as smoothly as BiJD. Both methods suffer from loss of track in iteration 13. If the tolerance is decreased to $\epsilon = 10^{-10}$, both SADPA and BiJD require the switch in order to converge to the desired accuracy. Extensive numerical experiments with many different dynamical systems has led to the insight that these problems specifically arise if there are (many) poles with relatively small real part, compared to the imaginary part. Apparently it is difficult to compute the real part of the pole (and corresponding eigenvectors) accurately, and errors in already found eigentriplets spoil the rest of the process. The PEEC example [28] in Figure 3.4 has a lot of dominant poles λ_i with $\mathrm{Im}(\lambda_i)$ of $O(1)$, while $\mathrm{Re}(\lambda_i)$ varies from $O(10^{-2})$ to $O(10^{-6})$.

If the (correction) equations are solved up to a certain precision, as will be discussed in the next section, then the above is still true and the use of subspace acceleration even helps more to converge to eigentriplets. The method of choice is then inexact BiJD.

3.6 Inexact variants for computing dominant poles

Inexact Rayleigh quotient iteration may have poor or even no convergence if the solutions of the linear systems are too inaccurate, see for instance [154, experiment 2.1]. Because the Jacobi-Davidson method computes the expansion vector as the

solution of a correction equation, it suffers less from this problem [74, 140, 154] and is therefore the method of choice for large-scale eigenvalue problems.

It can be shown, as a generalization of [154, Proposition 2.2], [143, Corollary 4.3], [74, Theorem 4.2] and Theorem 4.2 in Chapter 2, that inexact DPA also converges asymptotically quadratically. However, an inexact version of (subspace accelerated) DPA is expected to have the same, and probably even more severe problems than inexact RQI, as will be explained next. Inexact SADPA solves the equations in step 3 and 4 of Algorithm 3.2 with only moderate accuracy, for instance using a fixed number of iterations of GMRES (solving both equations separately) or BiCG (solving both equations simultaneously, see [74]). Suppose now that already a good approximation $\tilde{\lambda} \approx \lambda$ of the dominant pole is available, but that the approximate right and left eigenvectors are rather inaccurate. Since $\tilde{\lambda}$ will not change very much during the following iterations, also the expansion vectors will remain the same, because the right-hand sides remain constant. This explains why the problems might be even more severe for inexact DPA than for inexact two-sided RQI. Of course, increasing the number of iterations of the iterative solver as $\tilde{\lambda}$ converges might help, as is usual practice for inexact Newton schemes (see for instance [51, 141]), but still the ill-conditioning of the system as $\tilde{\lambda} \to \lambda$ causes difficulties, like in two-sided RQI. Since the projected correction equations in the two-sided Jacobi-Davidson method handle these problems better, BiJD is the preferred method when exact solves are not affordable. In practice, this better convergence behavior compensates the locally linear convergence [74, Theorem 5.3] of BiJD by far.

3.6.1 Preconditioning

Because the domain and image spaces of the operators in the correction equations (3.4.1)–(3.4.4) are not the same, it is unattractive to solve the equations with a Krylov solver. The usual way to fix this problem is to precondition with a suitable preconditioner (see for instance [51, Section 3.3] and [50, Section 6.2.4]). In the bi-E-orthogonal case (equations (3.4.1) and (3.4.2)), an obvious preconditioner for the right correction equation (3.4.1) is, given a preconditioner $K \approx A - \theta E$,

$$\tilde{K} = (I - E\mathbf{v}\mathbf{w}^*)K(I - \mathbf{v}\mathbf{w}^*E) \ : \ (E^*\mathbf{w})^\perp \longrightarrow \mathbf{w}^\perp,$$

with $\mathbf{w}^*E\mathbf{v} = 1$. Typical use in a Krylov solver requires actions of the form: solve $\mathbf{s} \perp (E^*\mathbf{w})$ from

$$(I - E\mathbf{v}\mathbf{w}^*)K(I - \mathbf{v}\mathbf{w}^*E)\mathbf{s} = \mathbf{t}, \quad \mathbf{w}^*\mathbf{t} = 0.$$

To obtain an expression for $\mathbf{s}$ (see also, for instance, [141, Section 3.2] and [50, Section 6.2.4]), first note that $\mathbf{w}^*E\mathbf{s} = 0$ and hence, since K is assumed to be invertible,

$$\mathbf{s} = K^{-1}\mathbf{t} + K^{-1}E\mathbf{v}(\mathbf{w}^*K\mathbf{s}).$$

Again use $\mathbf{w}^*E\mathbf{s} = 0$ to obtain by multiplication by $\mathbf{w}^*E$

$$\mathbf{w}^*K\mathbf{s} = -(\mathbf{w}^*EK^{-1}E\mathbf{v})^{-1}\mathbf{w}^*EK^{-1}\mathbf{t},$$

and with $\mathbf{q}_r = E^*\mathbf{w}$, $\mathbf{y}_r = K^{-1}E\mathbf{v}$ and $h_r = \mathbf{q}_r^*\mathbf{y}_r$ the solution $\mathbf{s}$ becomes, if h_r is nonzero,

$$\mathbf{s} = (I - \mathbf{y}_r h_r^{-1}\mathbf{q}_r^*)K^{-1}\mathbf{t}.$$

The following identities are easily verified by expanding the products (cf. [50, lemma 6.1]):

$$\begin{aligned}(I - \mathbf{y}_r h_r^{-1}\mathbf{q}_r^*)(I - \mathbf{v}\mathbf{w}^*E) &= (I - \mathbf{v}\mathbf{w}^*E)\\ (I - \mathbf{v}\mathbf{w}^*E)(I - \mathbf{y}_r h_r^{-1}\mathbf{q}_r^*) &= (I - \mathbf{y}_r h_r^{-1}\mathbf{q}_r^*)\\ (I - \mathbf{y}_r h_r^{-1}\mathbf{q}_r^*)K^{-1}(I - E\mathbf{v}\mathbf{w}^*) &= (I - \mathbf{y}_r h_r^{-1}\mathbf{q}_r^*)K^{-1}\end{aligned}$$

To conclude, using these identities, the (left) preconditioned right correction equation in the bi-E-orthogonal case becomes

$$(I - \mathbf{y}_r h_r^{-1}\mathbf{q}_r^*)K^{-1}(A - \theta E)(I - \mathbf{y}_r h_r^{-1}\mathbf{q}_r^*)\mathbf{s} = -(I - \mathbf{y}_r h_r^{-1}\mathbf{q}_r^*)K^{-1}\mathbf{r}_v,$$

with $Q^*\mathbf{s} = \mathbf{w}^*E\mathbf{s} = 0$, and clearly the operator maps $(E^*\mathbf{w})^\perp$ onto $(E^*\mathbf{w})^\perp$. In a similar way, the (left) preconditioned left correction equation becomes

$$(I - \mathbf{y}_l h_l^{-1}\mathbf{q}_l^*)K^{-*}(A - \theta E)^*(I - \mathbf{y}_l h_l^{-1}\mathbf{q}_l^*)\mathbf{s} = -(I - \mathbf{y}_l h_l^{-1}\mathbf{q}_l^*)K^{-*}\mathbf{r}_w,$$

with $\mathbf{v}^*E^*\mathbf{s} = 0$ and where $\mathbf{q}_l = E\mathbf{v}$, $\mathbf{y}_l = K^{-*}E^*\mathbf{w}$, and $h_l = \mathbf{q}_l^*\mathbf{y}_l$. Note that in this case K^* was taken as preconditioner of $(A - \theta E)^*$, but that other choices are possible as well.

If the search spaces are kept orthogonal, the (left) preconditioned right correction equation (3.4.3) becomes

$$(I - \mathbf{y}_r h_r^{-1}\mathbf{q}_r^*)K^{-1}(A - \theta E)(I - \mathbf{y}_r h_r^{-1}\mathbf{q}_r^*)\mathbf{s} = -(I - \mathbf{y}_r h_r^{-1}\mathbf{q}_r^*)K^{-1}\mathbf{r}_v,$$

with $\mathbf{v}^*\mathbf{s} = 0$ and where $\mathbf{q}_r = \mathbf{v}$, $\mathbf{y}_r = K^{-1}E\mathbf{v}$, and $h_r = \mathbf{q}_r^*\mathbf{y}_r$. Similarly, the (left) preconditioned left correction equation (3.4.4) becomes

$$(I - \mathbf{y}_l h_l^{-1}\mathbf{q}_l^*)K^{-*}(A - \theta E)^*(I - \mathbf{y}_l h_l^{-1}\mathbf{q}_l^*)\mathbf{s} = -(I - \mathbf{y}_l h_l^{-1}\mathbf{q}_l^*)K^{-*}\mathbf{r}_w,$$

with $\mathbf{w}^*\mathbf{s} = 0$ and where $\mathbf{q}_l = \mathbf{w}$, $\mathbf{y}_l = K^{-*}E^*\mathbf{w}$, and $h_l = \mathbf{q}_l^*\mathbf{y}_l$.

Similar expressions can be derived for right preconditioned corrections equations [50, Section 6.2.4].

Unlike the equivalence of exact SADPA and BiJD in Section 3.4.2, there is no relation between inexact SADPA and BiJD similar to [74, proposition 5.5], where it is shown that BiJD and subspace accelerated two-sided RQI are equivalent if the (correction) equations are solved with m and $m + 1$ steps of BiCG, respectively.

3.6.2 Deflation

If $X \equiv X_k$ and $Y \equiv Y_k$ have as their columns the k found right and left eigenvectors of (A, E) and are normalized so that $Y_k^*EX_k = I$, then the pencil

$$((I - EXY^*)A(I - XY^*E), (I - EXY^*)E(I - XY^*E))$$

has the same eigentriplets as (A, E), but with the eigenvalues corresponding to the found eigenvectors transformed to 0. Following the steps in Section 3.6.1, with deflation, the preconditioned right correction equation for the bi-E-orthogonal case becomes (if H_r is nonsingular)

$$(I - Y_r H_r^{-1} Q_r^*) K^{-1} (A - \theta E)(I - Y_r H_r^{-1} Q_r^*)\mathbf{s} = -(I - Y_r H_r^{-1} Q_r^*)\mathbf{r}_v, \quad (3.6.1)$$

with $Q_r^*\mathbf{s} = 0$, and where $Q_r = [E^*Y, E^*\mathbf{w}]$, $Y_r = K^{-1}[EX, E\mathbf{v}]$ and $H_r = Q_r^* Y_r$. Similar expressions can be derived for the left correction equation and for the correction equations for the orthogonal case.

3.6.3 Computational notes

For each of the preconditioned correction equations, the matrices Q, Y, and H grow as the number of found eigentriplets increases. In the case of the right correction equation for the bi-E-orthogonal case (3.6.1), for example, this can be handled efficiently by storing $Q_r^k = E^*Y$ and $Y_r^k = K^{-1}EX$, so that every iteration only the last column $\mathbf{q}_r = E^*\mathbf{w}$ of Q_r and $\mathbf{y}_r = K^{-1}E\mathbf{v}$ of Y_r need to be computed. If additionally $H_r^k = Q_r^{k*} Y_r^k$ is stored, H_r can every iteration be computed as

$$H_r = \begin{bmatrix} H_r^k & Q_r^{k*}\mathbf{y}_r \\ \mathbf{q}_r^* Y_r^k & \mathbf{q}_r^*\mathbf{y}_r \end{bmatrix}.$$

Also, if the approximate solution $\tilde{\mathbf{s}}$ is computed with a Krylov solver and the initial guess satisfies $Q^*\mathbf{s}_0 = 0$, then $Q^*\tilde{\mathbf{s}} = 0$ as well. Applications with the preconditioned operator reduce to

$$(I - YH^{-1}Q^*)K^{-1}(A - \theta E)\mathbf{s}.$$

3.6.4 Numerical experiment

To illustrate the better performance of inexact BiJD, Figure 3.5 shows the convergence histories of inexact BiJD and SADPA for the BIPS system of Section 3.7. The linear (correction) equations were solved using 10 steps of GMRES, with preconditioner $LU = s_0 E - A$, where $s_0 = 1i$ was the initial shift. Both methods started with $\mathbf{v}_1 = (s_0 E - A)^{-1}\mathbf{b}$ and $\mathbf{w}_1 = (s_0 E - A)^{-*}\mathbf{c}$, and were restarted every $k_{max} = 10$ iterations, keeping the $k_{min} = 2$ most dominant approximates. Inexact BiJD finds the first pole $\lambda_1 \approx -0.03353 \pm 1.079i$ within a few iterations and eventually also finds a second dominant pole $\lambda_2 \approx -0.5208 \pm 2.881i$. Inexact SADPA, on the other hand, was not able to approximate λ_1 and corresponding eigenvectors up to tolerance $\epsilon = 10^{-8}$. The search spaces were kept bi-E-orthogonal. Similar experiments with orthogonal search spaces showed worse performance for both BiJD and SADPA.

Where BiJD converges more or less smoothly, inexact SADPA suffers from stagnation and hampering around $\|\mathbf{r}\|_2 \approx 10^{-5}$: this can be explained by the fact that the projected operators in the correction equations are better conditioned than

the operators in the linear DPA equations, especially if $\theta \to \lambda$. In that case, the condition number of $\theta E - A$ becomes unbounded, while the (effective) condition number of the projected operator remains bounded (see also [74, p. 159]). As a consequence, preconditioned GMRES may compute better approximate solutions of the correction equations. This advantageous behavior of the Jacobi-Davidson method is well known, see [51, 74, 139, 140, 154] for more details and options to optimize the JD process.

Note that most advantage is taken from the projected corrections equations in the final phase of convergence to an eigentriplet. In the initial phase, subspace acceleration and the initial shift help to create promising search spaces, and it is not expected (and not confirmed by experiments) that it makes a big difference whether the BiJD corrections equations or the DPA equations are solved, since both are solved only up to moderate accuracy. To take care of the ill conditioning in the final phase of convergence, it is not only more efficient to use the BiJD corrections equations, but also more natural (correspondence with two-sided Rayleigh quotient iteration) than to use projected variants of the DPA equations.

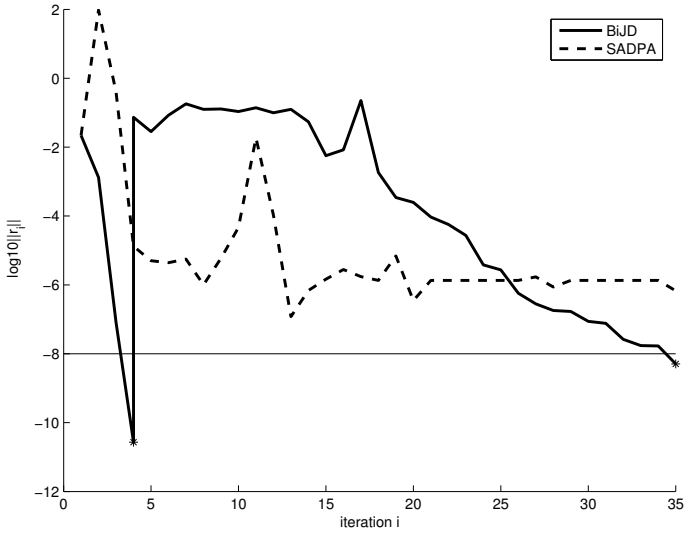

Figure 3.5: Convergence histories for inexact SADPA (dashed) and BiJD (solid), for the BIPS system of order $n = 13,251$. The linear equations were solved using 10 steps of GMRES with $LU = iE - A$ as preconditioner.

3.7 Numerical results

SADPA was tested on a number of systems, for a number of different input and output vectors **b** and **c**. Here the results for the Brazilian Interconnected Power

System (BIPS) are shown (see Section 3.3.2 for the application of SADPA to a small test system). The system data corresponds to a year 1999 planning model, having 2,400 buses, 3,400 lines, a large HVDC link, 123 power plants with detailed dynamic representation, 46 of which have power system stabilizers. The BIPS model is linearized around an operating point having a total load of 46,000 MW, with the North-Northeast generators exporting 1,000 MW to the South-Southeast Region, through the planned 500 kV, series compensated North-South intertie. The Power Oscillation Damping (POD) controllers of the two Thyristor Controlled Series Compensators (TCSC) are disabled, causing the low frequency North-South mode to become poorly damped ($\lambda_{ns} \approx -0.0335 + i1.0787$).

In the experiments, the convergence tolerance used was $\epsilon = 10^{-10}$. The spaces V and W were limited to dimension $n \times 10$ (i.e. a restart, with $k_{min} = 4$, every $k_{max} = 10$ iterations). The results are compared with the results of DPSE on quality of the modal equivalents, computed poles, CPU time and number of factorizations. DPSE uses deflation of complex conjugate pairs. All experiments were carried out in Matlab 7.3 [151] on a SUN Ultra 20 (AMD Opteron 2.8GHz, 2GB RAM).

The state-space realization of the BIPS model has 1664 states. The sparse, unreduced Jacobian has dimension 13251. Like the experiments in [91, 90], the practical implementation operates on the sparse unreduced Jacobian of the system, instead of on the dense state matrix A.

To demonstrate the performance of SADPA, it is applied to six transfer functions of BIPS to compute a number of dominant poles (complex conjugate pairs are counted as one pole). The first four transfer functions relate the rotor shaft speed deviations (W) of major synchronous generators to disturbances applied to the voltage references of their corresponding excitation control systems. Note that only a single shift, $s_1 = 1i$, is used: after the first pole has converged, the next most dominant approximate pole is used as new shift. The results are in Table 3.2 and the Bode plots of the corresponding modal equivalents, complete models and errors for the first four transfer functions are in Figure 3.6 and Figure 3.7. It is clearly observable that the reduced models capture the important dynamics. For completeness, the 15 most dominant poles and corresponding residues of $W_{6405}/Vref_{6405}$, according to the index $|R_i|/|\mathrm{Re}(\lambda_i)|$, are shown in Table 3.3.

It can also be observed in Table 3.2 that the $\|\hat{\mathbf{y}}_i\|_2 = \|\hat{\mathbf{x}}_i\|_2 = 1$ scaling leads to much better results than the $\hat{\mathbf{y}}_i^* E \hat{\mathbf{x}}_i = 1$ scaling. The decrease in the number of LU-factorizations and CPU time is significant. Although not shown here, both approaches find the most dominant poles (in a different order), and the modal equivalents are of comparable quality (except for $W_{6405}/Vref_{6405}$ and $W_{1107}/Vref_{1107}$, where 10 additional poles are needed). When using the scaling $\|\hat{\mathbf{y}}_i\|_2 = \|\hat{\mathbf{x}}_i\|_2 = 1$, the selection is based on the angles $\angle(\hat{\mathbf{x}}_i, \mathbf{c})$ and $\angle(\hat{\mathbf{y}}_i, \mathbf{b})$, which is in accordance with the results in Chapter 2. In the addendum of Chapter 4 a more detailed comparison is made.

In Table 3.3 there are also indications whether SADPA, DPSE and DPSE with deflation (DPSEd) were able to find the most dominant poles when given at most 35 seconds to compute 20 dominant poles. In Table 3.4, SADPA, DPSE and DPSE with deflation (DPSEd) are compared on computational time. SADPA succeeds in

Table 3.2: Results of SADPA for six transfer functions of the Brazilian Interconnected Power System (BIPS), with two types of scaling. Shift $s_1 = 1i$.

Transfer function	#poles	$\hat{\mathbf{y}}_i^* E \hat{\mathbf{x}}_i = 1$		$\|\hat{\mathbf{y}}_i\|_2 = \|\hat{\mathbf{x}}_i\|_2 = 1$	
		#LU	Time (s)	#LU	Time (s)
$W_{5061}/Vref_{5061}$	30	846	155	169	42
$W_{6405}/Vref_{6405}$	45	852	164	222	58
$W_{1155}/Vref_{1155}$	40	2229	462	256	65
$W_{1107}/Vref_{1107}$	40	816	158	217	54
P_{sc}/B_{sc}	50	2047	446	289	78
$Pt_{501}/Pref_{501}$	70	3436	855	317	87

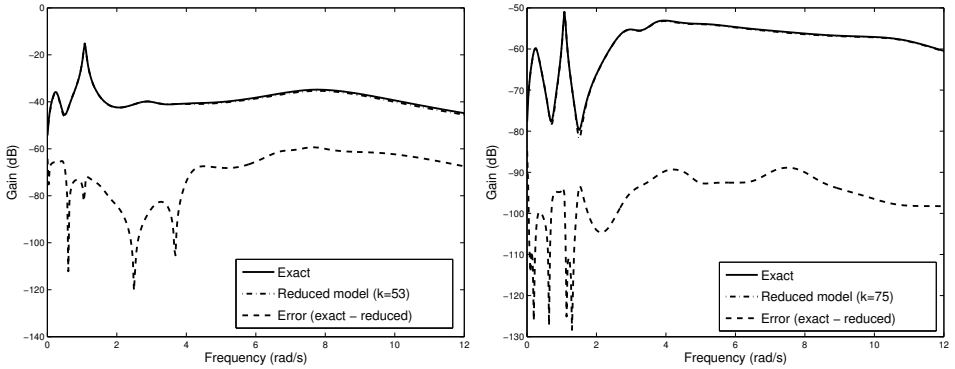

Figure 3.6: Bode plots of modal equivalent, complete model and error for transfer function $W_{5061}/Vref_{5061}$ (left, $k = 53$ states) and $W_{6405}/Vref_{6405}$ (right, $k = 75$ states) of BIPS (1664 states in the complete model).

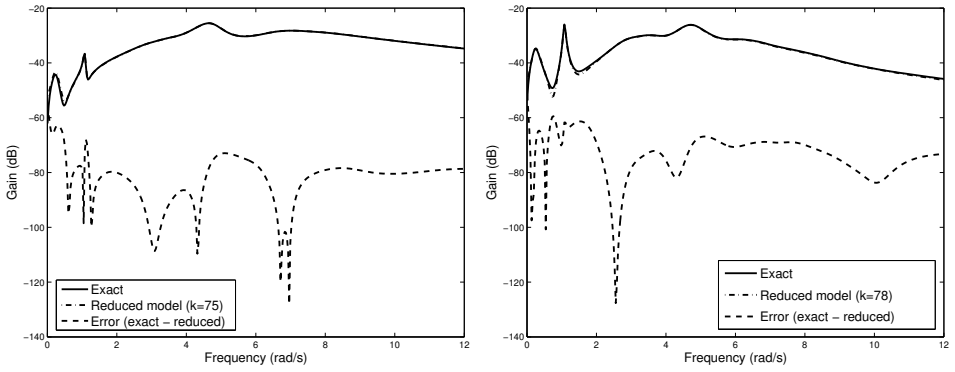

Figure 3.7: Bode plot of modal equivalent, complete model and error for transfer function $W_{1155}/Vref_{1155}$ (left, $k = 75$ states) and $W_{1107}/Vref_{1107}$ (right, $k = 78$ states) of BIPS (1664 states in the complete model).

Table 3.3: The 15 most dominant poles and corresponding residues of $W_{6405}/Vref_{6405}$, sorted in decreasing $|R_i|/|\mathrm{Re}(\lambda_i)|$ order (29-state modal equivalent). Poles found by SADPA, DPSE, and DPSEd are marked with a +.

Num. Mode	Modes	Residues		SADPA found	DPSE found	DPSEd found
		Magnitude	Phase			
1	$-0.0335 \pm 1.0787i$	$9.241 \cdot 10^{-5}$	± 173.0	+	+	+
3	$-0.5208 \pm 2.8814i$	$7.542 \cdot 10^{-4}$	± 139.7	+	+	+
5	$-0.5567 \pm 3.6097i$	$7.475 \cdot 10^{-4}$	± 102.9	+	+	+
7	$-2.9445 \pm 4.8214i$	$3.022 \cdot 10^{-3}$	± 167.7	+	+	+
9	$-0.1151 \pm 0.2397i$	$1.078 \cdot 10^{-4}$	± 176.9	+	+	+
11	$-6.4446 \pm 0.0715i$	$5.162 \cdot 10^{-3}$	± 144.0	+		+
13	-4.9203	$3.386 \cdot 10^{-3}$	0	+		
14	$-7.5118 \pm 0.2321i$	$3.625 \cdot 10^{-3}$	± 128.7	+		
16	$-2.3488 \pm 11.001i$	$1.014 \cdot 10^{-3}$	± 4.167	+	+	+
18	$-10.068 \pm 1.1975i$	$3.841 \cdot 10^{-3}$	± 17.57	+		+
20	$-1.4595 \pm 10.771i$	$5.305 \cdot 10^{-4}$	± 47.01	+	+	+
22	$-20.539 \pm 1.0930i$	$6.224 \cdot 10^{-3}$	± 65.00	+		
24	$-2.8052 \pm 11.551i$	$7.558 \cdot 10^{-4}$	± 8.385	+	+	
26	$-4.0233 \pm 4.2124i$	$1.009 \cdot 10^{-3}$	± 35.78	+	+	
28	$-0.7584 \pm 4.9367i$	$1.780 \cdot 10^{-4}$	± 146.0	+	+	+

Table 3.4: Computational statistics for SADPA ($s_1 = 1i$), the DPSE and the DPSE with deflation (DPSEd) (shifts $1i, 1.5i, \ldots, 10.5i$), when computing 20 dominant poles.

Method	#LU	time (s)	#poles	#repeated poles
SADPA	204	35	20	0
DPSE	160	25	20	3
DPSEd	193	35	17	0

finding both real and complex dominant poles, and, moreover, finds all of the 15 most dominant poles, while DPSE and DPSEd miss 5 of the most dominant poles. DPSE also recomputed already found poles three times. To make the comparison as fair as possible, the restart parameters for SADPA were $k_{min} = 1$ and $k_{max} = 10$, and the $\hat{\mathbf{y}}_i^* E \hat{\mathbf{x}}_i = 1$ scaling was used (SADPA was even faster for $k_{min} = 4$ and $\|\hat{\mathbf{y}}_i\|_2 = \|\hat{\mathbf{x}}_i\|_2 = 1$ scaling).

DPSE has more difficulties to converge as the number of shifts increases: typically, the tolerance is not reached for an increasing number of poles (like in SADPA, this can be (and was in the experiments) fixed by doing an additional step of two-sided Rayleigh quotient iteration). Besides that, also the computational costs increase rapidly as the number of shifts increases, because the interaction matrices grow. To compute 60 poles with DPSE, it has to be started again with different sets of shifts, which has the risk of computing the same poles again, is time consuming and requires human interaction. SADPA, however, finds 60 dominant poles, starting with just one single shift, without any human interaction during the process. Because of the selection strategy, truly dominant poles are computed; the deflation

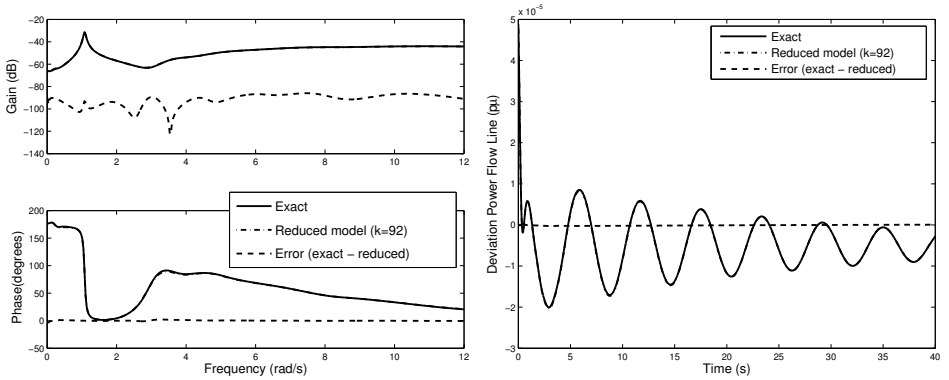

Figure 3.8: Bode plot (left) and step response (right) of modal equivalent, complete model and error for transfer function $P_{sc}(s)/B_{sc}(s)$ of BIPS (92 states in the modal equivalent, 1664 in the complete model).

strategy prevents repeated computation of the same pole. Furthermore, SADPA succeeds in computing real poles, while DPSE has many difficulties in computing real poles: real shifts are needed for DPSE, while SADPA finds the real poles automatically. SADPA is not sensitive to the initial shift: repeated experiments with other shifts give the same results.

The fifth transfer function of BIPS is P_{sc}/B_{sc}, relating the active power deviations flowing through the North-end series capacitor of the planned intertie, to disturbances in the reference value of the TCSC series admittance. This transfer function was used in the basic design of the POD controllers of the two TCSCs, in order to damp the North-South mode [32, 60, 94]. Modal equivalents of the transfer function for damping the North-South mode, whose state-space realization has a direct transmission term $d = 4.88 \cdot 10^{-3}$, are considered in [94]. Figure 3.8 (left) shows the frequency response of the complete model and the reduced model (92 states) together with the error. Figure 3.8 (right) shows the corresponding step response (step 0.01)[2]. The North-South mode (1 rad/s = 0.17 Hz) is well observable in both responses, and the reduced model nicely captures the system oscillations. The reduced model (50 poles, 92 states) was computed by SADPA in 78 seconds (289 factorizations). Occasional corruption of the search spaces by components in the direction of eigenvectors corresponding to eigenvalues at infinity was successfully limited by keeping these components in the search spaces at restarts. A table with the dominant poles and corresponding residues can be found in [94].

The sixth transfer function, $Pt_{501}/Pmec_{501}$, relates the active power deviations of a large hydro-electric plant, located in the Southeast region, to disturbances applied to its speed-governor reference. For this transfer function it is known from numerical experiments that DPSE has difficulties in producing a good modal equivalent: several DPSE attempts are needed to obtain an acceptable modal equivalent.

[2]If $h_k(t)$ is the inverse Laplace transform of $H_k(s)$ (3.2.4), the step response for step $u(t) = c$ of the reduced model is given by $y(t) = \int_0^t h(t)u(t) = c(\sum_{i=1}^k \mathrm{Re}((\frac{R_i}{\lambda_i}(\exp(\lambda_i t) - 1)) + d)$.

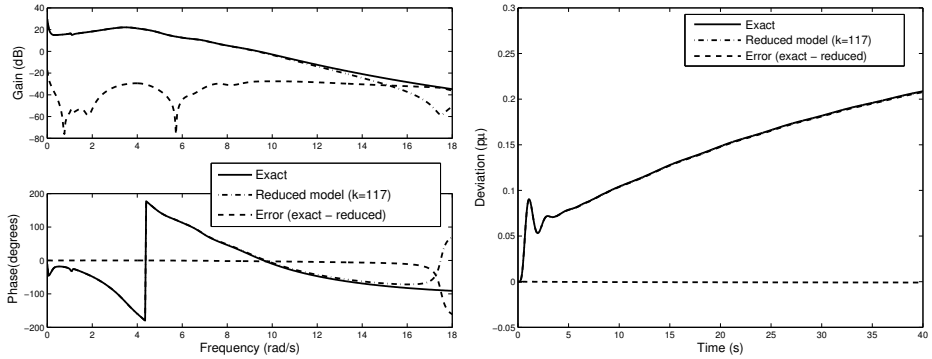

Figure 3.9: Bode plot (left) and step response (right) of modal equivalent, complete model and error for transfer function $Pt_{501}(s)/Pref_{501}(s)$ of BIPS (117 in the modal equivalent, 1664 in the complete model).

SADPA is able to *automatically* produce good results for both the frequency and step response, as can be observed from Figure 3.9. SADPA computed the reduced model (70 poles, 117 states) in 87 seconds (317 factorizations) using just a single initial shift $s_1 = 1i$, in one single run.

It should be noted that all modal equivalents in this section can be reduced even further by neglecting less dominant contributions, or by application of balanced truncation to a state-space realization of the modal equivalent.

3.8 Reflection

SADPA is primarily an algorithm for the computation of dominant poles of large-scale transfer functions. The corresponding left and right eigenvectors, that are computed as well by SADPA, can be used to construct reduced-order models of the original system in the form of modal equivalents. Although these modal equivalents are of practical value, both for other (transient) simulations with and to verify (via the Bode plot) whether all dominant poles are found, there are two points of concern in comparison with Krylov subspace projection based model reduction techniques that immediately come to front: first, in exact SADPA, every iteration exact solves with $s_k E - A$ and $(s_k E - A)^*$ are required (via a single LU-factorization), and second, there is no moment matching property for the modal equivalents produced by SADPA. The first point can be overcome by using inexact solves, as described in Section 3.6, but in this section it is assumed that exact solves are feasible (recall that SADPA is the method of choice in this case). This assumption is reasonable, since Krylov based methods also require at least exact solves with $\sigma_0 E - A$ for one σ_0 (usually via LU-factors). Instead of matching moments, the poles of the modal equivalent are poles of the original system [18] and hence modal approximation preserves stability automatically and is applicable to both stable and unstable systems. Although the exact location (except for the sign of the real part) of poles is not important in all kinds of analysis of dynamical

systems, it is important in for instance vibration analysis [18], where the poles correspond to resonance frequencies of the original system, and damping control of electromechanical oscillations in power systems [90, 91, 93], where stabilizers are designed to damp specific dominant poles.

The goal of this section is not to argue which one of modal approximation and Krylov based methods is the best, but to show the qualities and limitations of both methods, and how one method may help the other to deal with or even overcome its limitations. Krylov based methods are widely described in the literature, see for instance [5, part IV] and references therein. Modal approximation methods, on the other hand, are often mentioned briefly or even not, and if mentioned, focus is usually on the limitations (see for instance [5, Section 9.2]). It may be the case that this typical focus is caused by the availability of many Krylov based schemes such as PRIMA [105] and rational Arnoldi and Lanczos, while practical modal approximation methods, other than computing a *full* eigendecomposition and selecting the dominant eigenvectors [159], were not available. SADPA is novel in the sense that it computes the dominant poles measured in (scaled) residue norm, instead of nearness to the imaginary axis, as is done in some literature [5, p. 283], and that it computes these poles automatically in a single run, together with the left and right eigenvectors, that can be used to construct the modal equivalent. SADPA succeeds in finding the dominant poles even if good initial estimates are not available and does not require a full (block) diagonalization of (A, E). This allows for a comparison between modal approximation by SADPA and Krylov based methods. Theoretical considerations are illustrated by numerical examples.

Of the Krylov subspace projection methods for model reduction, rational interpolation is probably one of the most effective techniques nowadays (see Section 1.5.3, and see [70] and [5] for good introductions). Contrary to the single interpolation point methods PRIMA [105] and PVL [48], rational interpolation uses multiple interpolation points and constructs bases for rational Krylov subspaces instead of ordinary Krylov subspaces. The big advantage of rational interpolation is that moments are matched around several distinct interpolation points, leading to much better and smaller reduced-order models than conventional Krylov based methods [58]. If good interpolation points are known (and the number of moments to match around each point), then the availability of efficient codes for the construction of ((bi)-orthogonal) bases for the rational Krylov subspaces makes rational interpolation an efficient and effective approach. If, on the other hand, there is no or limited knowledge of good interpolation points, rational interpolation becomes less effective. Although there are techniques for dynamically selecting and adding interpolation points, see [58] and [70, Chapter 6], rational interpolation may not succeed in capturing (some of) the dominant behavior, typically visible in missing peaks in the Bode plot.

These problems are inherent to the use of Padé approximations and (rational) Krylov subspaces [58, p. 39]. Padé approximations are exact at the point of interpolation σ, while accuracy is lost away from σ, even more rapidly if σ is close to a pole [14, 30]. The methods used to construct the bases for the rational Krylov subspaces are rational Bi-Lanczos [58] and dual rational Arnoldi [70, Section 4.1.2].

These methods typically tend to approximate the eigenvalues at the outer edge of the spectrum of $(A-\sigma E)^{-1}E$, converging to well-separated eigenvalues at the edge first. Practically speaking this means that the interpolation points should be chosen in such a way that the dominant poles become well separated eigenvalues at the edge of the spectrum of $(A-\sigma E)^{-1}E$. Without knowledge of the location of the dominant poles, probably the best one can do is to logarithmically space the real interpolation points between ω_{min} and ω_{max} to obtain the general frequency response, and to add purely imaginary interpolation points to capture the exact response at frequencies of interest, as proposed in [70, Chapter 6]: real interpolation points σ tend to transform the eigenvalues $|\lambda_i| \approx \sigma$ to the outer edge of the spectrum of $(A-\sigma E)^{-1}E$, leading to reduced models that capture the global response, while purely imaginary interpolation points σ transform eigenvalues $\lambda_i \approx \sigma$ to the outer edge of spectrum, leading to locally good results around frequencies $\omega \approx \text{imag}(\sigma)$.

Based on the properties of the rational Krylov subspace based methods, one may expect that these methods work well for transfer functions with relatively smooth frequency response: as is also confirmed by numerical experience, real interpolation points suffice in many cases to obtain good reduced-order models. As an illustration, consider the Bode plots of two reduced models of $W_{6405}/Vref_{6405}$ (see Section 3.7) in Figure 3.10. The reduced-order model computed by dual Arnoldi with a single interpolation point $\sigma_0 = 0$ is not able to capture the frequency response for the range of interest, even not for a model of order 85. Adding a second point $\sigma_1 = 1$ leads to a better result, but more interpolation points are needed to obtain an exact match. Although not shown here, reduced-order models of similar sizes produced by PRIMA are of worse quality, since PRIMA computes a basis for the Krylov space $\mathcal{K}^k((\sigma_0 E-A)^{-1}E, (\sigma_0 E-A)^{-1}\mathbf{b})$ and neglects the output vector $\mathbf{c} \neq \mathbf{b}$ (and matches only k moments, while the (two-sided) rational k-dimensional models match $2k$ moments, albeit at the cost of $2k$ matrix vector multiplications against k for PRIMA).

The situation becomes different if the frequency response shows a large number of peaks, caused by (relatively) dominant poles with small real part, close to each other, or if dominant poles are not transformed to the outer edge of the spectrum due to the choice of interpolation points, and consequently are not present in the reduced model. SADPA, on the other hand, is expected to be able to handle both smooth and less smooth transfer functions, since it computes the most dominant poles and constructs modal equivalents using the corresponding left and right eigenspaces. Inherent to modal approximation, however, is that the reduced model is accurate for frequencies in the neighborhood of the imaginary parts of the dominant poles, while less accurate for distant frequencies (typically the smooth parts of the response). The following two examples illustrate these issues and also indicate that rational Krylov methods and SADPA can be combined to deal with these problems.

The model of a portable compact disc player, where the control task is to point the lens of the laser to the track of pits of the rotating CD, is also considered in [58, 70] and is available via [28]. Figure 3.11 shows the frequency response for

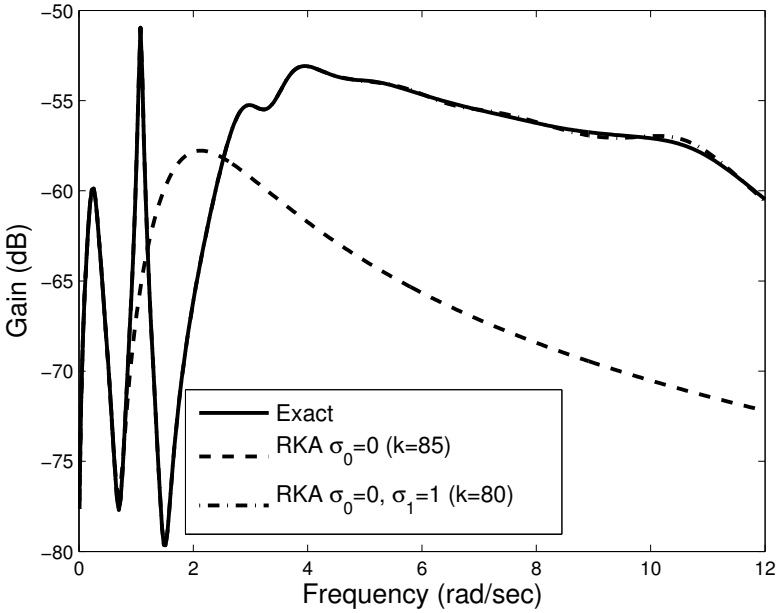

Figure 3.10: Exact frequency response of $W_{6405}/Vref_{6405}$, and responses of a 85-th order RKA model (interpolation point $\sigma_0 = 0$) and a 80-th order RKA model ($\sigma_0 = 0, \sigma_1 = 1$).

the second input and the first output (the 120-th order system has 2 inputs and 2 outputs), together with the responses of 60-th order reduced models computed by SADPA and dual rational Arnoldi (RKA, the implementation of [70, Appendix B] was used). SADPA needed 203 iterations (LU-factorizations) to compute 30 dominant poles and construct the real modal equivalent ($s_0 = 1i$, $k_{min} = 1$, $k_{max} = 10$, $1.4s$ CPU time), while RKA generated a 60-th order real model by matching 10 moments around each of the 12 logarithmically spaced points in the interval $[1, 5 \cdot 10^5]$ within $0.4s$ CPU time (to match p moments, $p/2$ left and right basis vectors are needed). Note that only a single initial estimate $s_0 = 1i$ was needed for SADPA. In the "eye-norm" there is hardly any difference: both frequency responses seem to match exactly. The relative error in Figure 3.12 shows a different picture. For frequency ranges where the response is relatively smooth or only has isolated peaks, the RKA model is orders of magniture more accurate. If there are many peaks close to each other, the SADPA model is more accurate, while the RKA response even has relative errors of order 1 (see near $\omega = 10^4$ rad/s). This example confirms that Krylov based models capture the global dynamics accurately while missing important details, and that modal equivalents are locally accurate, while only moderately accurate for frequencies away from the imaginary parts of dominant poles. It also suggests that the modal equivalent may be improved by adding a small part of the Krylov subspaces to left and right eigenspaces, and that RKA model may be improved by adding some of the left and right eigenvectors to the Krylov subspaces. These and other possibilities will be discussed in more

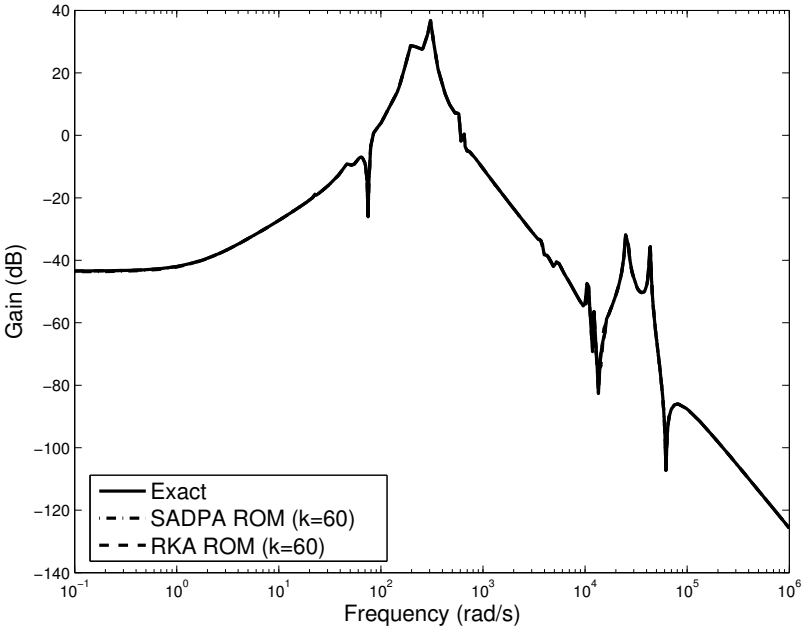

Figure 3.11: Exact frequency response of the CD player model, and respones of 60-th order approximations computed by SADPA (dash-dot) and dual rational Arnoldi (RKA, dash).

detail after the following example.

The PEEC 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 [28, 129, 161] for more details and the model data). The PEEC model is known as a difficult problem, because it has many poles close to each other [73]. The frequency response is shown in Figure 3.13, together with the frequency responses of a 89-th order model computed by SADPA, and a 90-th order real model computed by RKA. SADPA needed 222 iterations (LU-factorizations) to compute 45 dominant poles ($s_0 = 96i$, $k_{min} = 4$, $k_{max} = 10$, $38s$ CPU time), leading to a 89-th order reduced real model. RKA generated a 90-th order real model by matching 2 moments around each of the 90 logarithmically spaced imaginary points in the interval $i[0.1, 10^5]$ within $41s$ CPU time (82 states after removing dependencies in the bases [70, Section 3.3.2]). Compared to the results of the CD player model, three important observations can be made. Firstly, the frequency responses of the SADPA and RKA models differ from the exact response in the "eye-norm" (cf. $\omega > 200$ rad/s for SADPA and RKA, and $\omega \approx 8$ rad/s for RKA). Secondly, the relative errors (Figure 3.14) for SADPA and RKA differ only by 2 orders for smooth parts of the frequency response. Thirdly, the SADPA model is much more accurate in the interval $[1, 10]$ rad/s: in the "eye-norm" the match is exact, and the relative error is much smaller. As was also reported in [73, p. 146], it is difficult to improve the RKA model by using even more interpolation points. SADPA, on the other hand, finds the dominant poles automatically without any

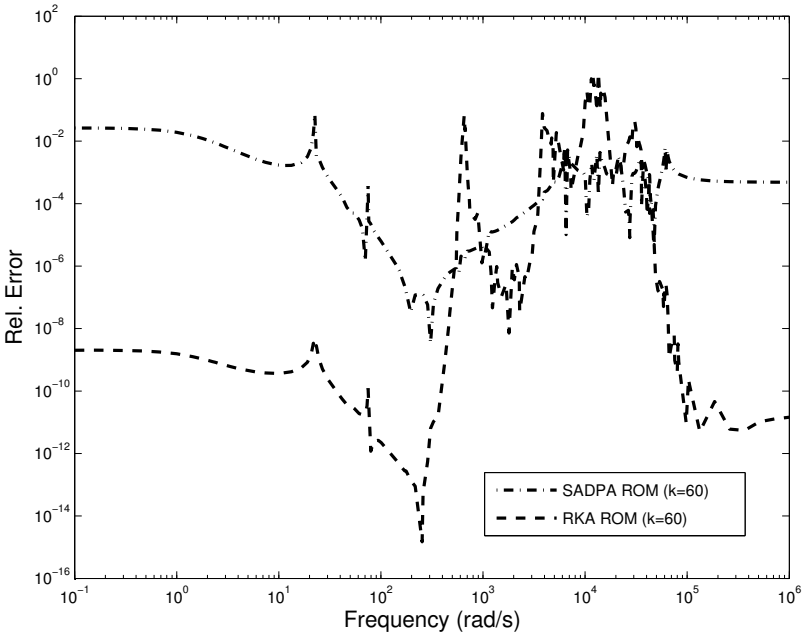

Figure 3.12: Relative error in the frequency response of the CD player model for the 60-th order approximations computed by SADPA (dash-dot) and dual rational Arnoldi (RKA, dash).

human interaction or a priori knowledge of the locations.

A straightforward approach to improve a modal equivalent computed by SADPA, is to (orthogonally) expand the (real) bases Y and X for the left and right dominant eigenspaces with a small number of basis vectors computed by the rational Krylov subspace method. If the columns of W and V form a basis for the left and right rational Krylov spaces, then the extended reduced model can be constructed by using $Z = [Y, W]$ and $Q = [X, V]$ in $(Z^*EQ, Z^*AQ, Z^*\mathbf{b}, Q^*\mathbf{c}, d)$. Note that this new model preserves the exact poles of the modal equivalent (that are exact poles of the original model as well), and also preserves the matched moments due to W and V, since $\text{span}(W) \subset \text{span}(Z)$ and $\text{span}(V) \subset \text{span}(Q)$. In Figure 3.15 the relative error for the original 60-th order SADPA model is shown together with the relative errors for two extended models. The 66-th order model was constructed by expanding the eigenspaces with six basis vectors for the rational Krylov subspaces with interpolation points $\{1, 10, 100, 10^3, 10^4, 10^5\}$. The 72-nd order model was constructed by expanding the eigenspaces with 6 six basis vectors for the rational Krylov subspaces with interpolation points $\{1, 5, 10, 50, 100, 500, 10^3, 5 \cdot 10^3, 10^4, 5 \cdot 10^4, 10^5, 5 \cdot 10^5\}$ (matching two moments around each point in both cases). Note that reduced models based on only these Krylov spaces are of poor quality. The improvement of the modal equivalent is apparent, and similar results can be reported for the PEEC model, although less dramatic, since the modal equivalent already has a small relative error. In practice, usually a naive choice for the interpolation points (logarithmically spaced) and 2

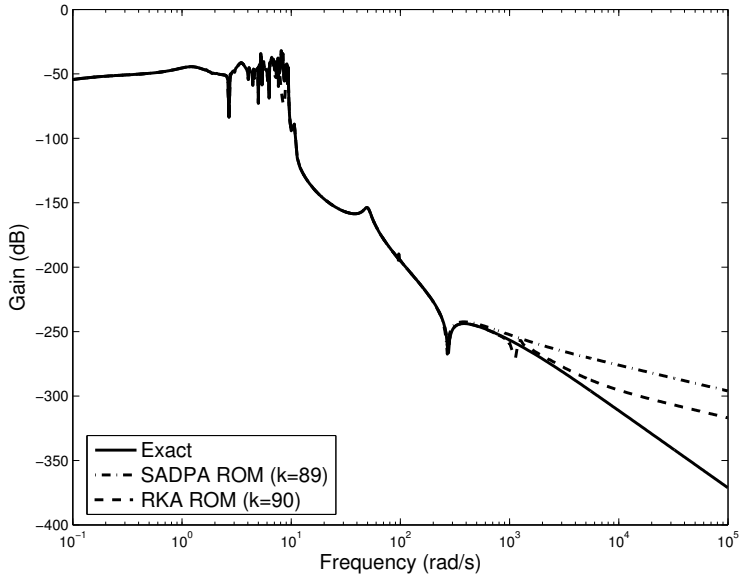

Figure 3.13: Exact frequency response of the PEEC model, and responses of 89-th order model computed by SADPA (dash-dot) and 90-th order model computed by dual rational Arnoldi (RKA, dash).

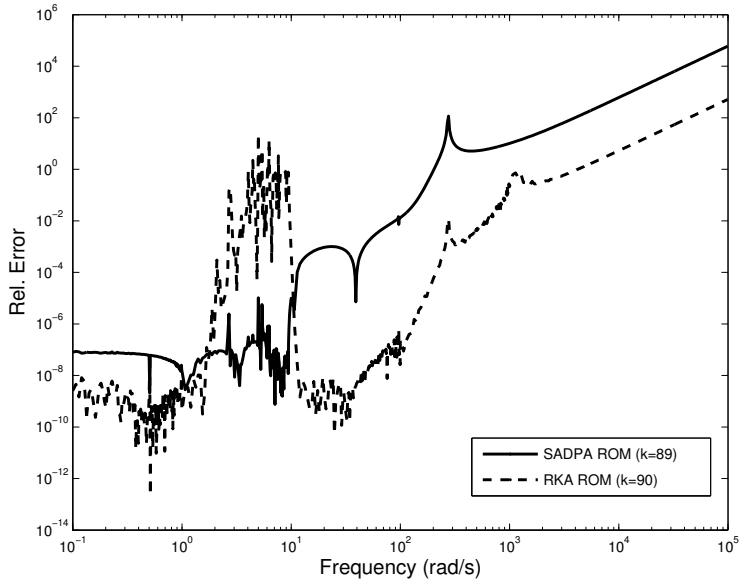

Figure 3.14: Relative error in the frequency response of the PEEC model for the 89-th order model computed by SADPA (solid) and 90-th order model computed by dual rational Arnoldi (RKA, dash)

matched moments per points suffice to improve the modal equivalent significantly: this will improve the global frequency response, while the local details are already covered by the dominant modes found by SADPA.

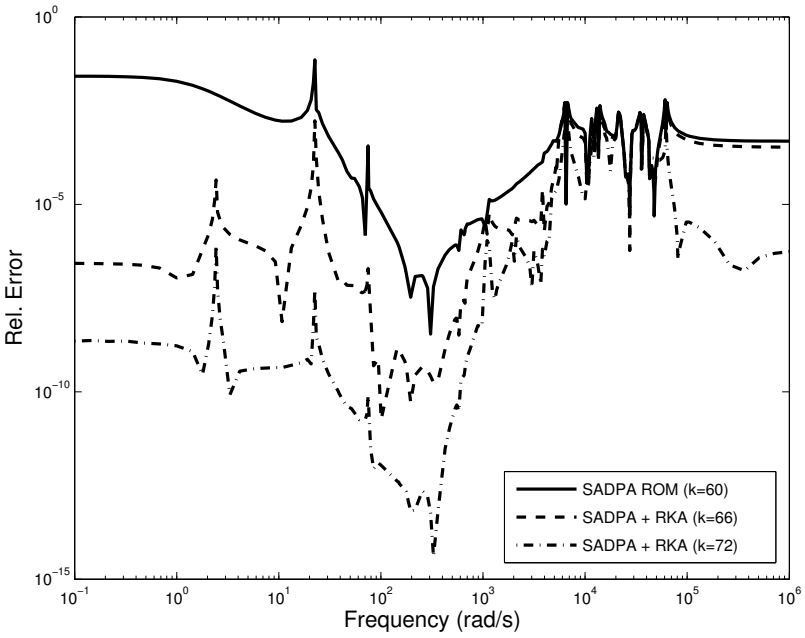

Figure 3.15: Relative error in the frequency response of the CD player model for the 60-th order approximation computed by SADPA (solid), and for 66-th (dash) and 72-th order (dash-dot) extended models, constructed by adding 6 and 12 dual rational Arnoldi basis vectors, respectively.

The other way around, reduced models constructed by rational Krylov methods can be improved by expanding the spaces with left and right eigenvectors corresponding to dominant poles. Again, this does not violate the moment matching properties and also leaves the found poles unchanged. This approach is useful if some isolated peaks are missed by the reduced model: the corresponding frequencies are good estimates for the missing poles and can be used as initial estimates for SADPA. For the CD player and PEEC models this approach is less practical, because most likely a large number of dominant poles has to be added, for which human interaction is needed. Vice versa, the imaginary parts of dominant poles computed by SADPA may be good interpolation points for the rational Krylov methods [70, Chapter 6]; an example of this can be found in Section 6.4.2 of Chapter 6. If exact solves are not feasible, inexact two-sided Jacobi-Davidson (Section 3.6) can be used to compute a few dominant poles. Yet another option is to use rational Krylov spaces as initial search spaces for SADPA.

The idea of improving modal equivalents by expanding the eigenspaces with rational Krylov spaces also sheds new light on the results in Section 3.7. As is also observed in experiments for other large-scale systems, SADPA tends to converge to the most dominant poles rather quickly, but stagnates when the remaining poles are

more or less equally dominant. In other words, SADPA smoothly computes modal equivalents that capture the peaks of the frequency response, but has difficulties in matching the response between peaks. These observations suggest to compute the most dominant poles with SADPA, and to expand the eigenspaces with relatively small rational Krylov spaces for a small number of interpolation points, as described before.

In Figure 3.16 the relative errors in the frequency response of $W_{6405}/Vref_{6405}$ are shown for an 18-th order RKA model (6 moments around each of the points $\{0, 2, 4, 6, 8, 10\}$), a 36-th order SADPA modal equivalent (20 (complex conjugate pairs of) poles), a 37-th order SADPA + RKA model (19 left and right eigenvectors (10 poles) + 18 RKA vectors), and 54-th order SADPA + RKA (36 left and right eigenvectors (20 poles) + 18 RKA vectors) model (cf. Figures 3.6 and 3.10). The two small models are not very accurate, although one can recognize that the SADPA model captures the details, while the RKA model makes a more global match. The combined models have the best of both and lead to acceptable reduced models, of better quality and computed within less time than the large SADPA models in Section 3.7, since the construction of the RKA bases requires only 4 seconds.

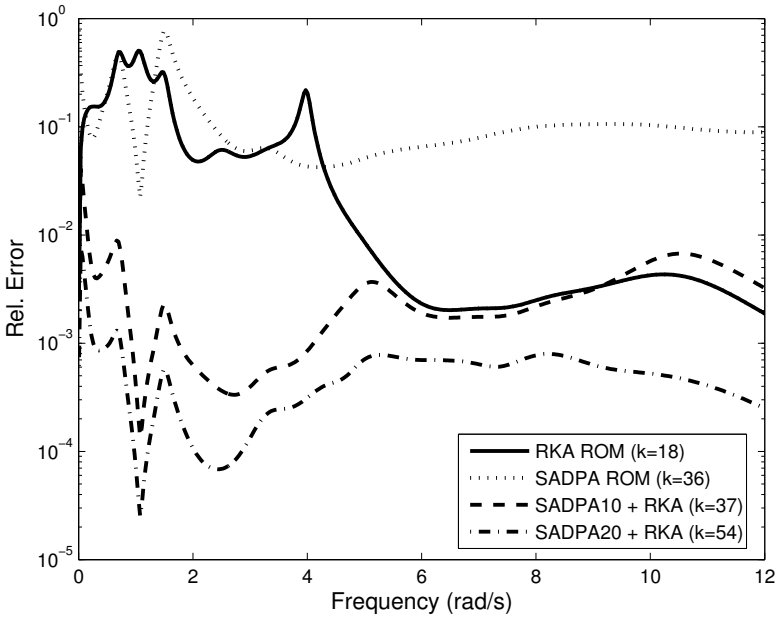

Figure 3.16: Relative errors in the frequency response of $W_{6405}/Vref_{6405}$ for 18-th order RKA model, 36-th order SADPA modal equivalent, 37-th order SADPA(10 poles, 19 states) + RKA(18) model and 54-th order SADPA(20,36) + RKA(18) model.

3.9 Conclusions

The algorithm described in this chapter, SADPA, is a fast method to compute the dominant poles and corresponding eigenvectors of a scalar transfer function. It has several advantages over existing methods. Firstly, it is more robust because it uses a natural selection method to converge to both real and complex dominant poles. Secondly, SADPA needs less iterations to converge by using subspace acceleration. Thirdly, it has less risk of missing a dominant pole, and of computing an already found pole, because of effective and efficient deflation techniques. Fourthly, SADPA is completely automatic: with a single initial estimate it is able to compute as many dominant poles as wanted, without intermediate human interaction.

It was also shown that if the linear (correction) equations are solved exactly, SADPA is equivalent to the two-sided Jacobi-Davidson method. If it is feasible to solve the linear (correction) equations exactly, using for instance LU-factorizations, then SADPA is the method of choice since the costs per iteration are less, and deflation can be done in an efficient and stable way via the input and output vectors (only a one time deflation is needed per found eigentriplet). If, on the other hand, exact solves are not feasible and one has to use iterative methods like GMRES, then two-sided Jacobi-Davidson is the preferred method, because the approximate solutions of the correction equations lead to better expansions of the search spaces and better convergence.

The modal equivalents constructed using the left and right eigenvectors of the dominant poles computed by SADPA, can be of use for further analysis of the original system. It was shown how rational Krylov methods can be used to improve the modal equivalents at low costs. The dominant poles computed by SADPA can be used to determine interpolation points for rational Krylov methods.

Addendum

The first version of SADPA (without the efficient deflation as described in Section 3.3.1) was presented in [123]

> Joost Rommes and Nelson Martins, *Efficient computation of transfer function dominant poles using subspace acceleration*, IEEE Transactions on Power Systems **21** (2006), no. 3, 1218–1226,

but this chapter is rewritten, reorganized and extended with new results, relations to Jacobi-Davidson and rational Krylov, additional numerical experiments, and reflections on rational Krylov based model reduction methods.