Universiteit Utrecht

Department
of Mathematics

Efficient computation of transfer function dominant poles using subspace acceleration

by

Joost Rommes and Nelson Martins

Preprint

nr. 1340

December, 2005

Efficient computation of transfer function dominant poles using subspace acceleration

Joost Rommes* and Nelson Martins†

December, 2005

Abstract

This paper describes a new algorithm to compute the dominant poles of a high-order scalar transfer function. The algorithm, called the Subspace Accelerated Dominant Pole Algorithm (SADPA), is more robust than existing methods in finding both real and complex dominant poles, and faster because of subspace acceleration. SADPA is able to compute the full set of dominant poles and produce good modal equivalents automatically, without any human interaction.

1 Introduction

Recent work on power system stability, controller design and electromagnetic transients has used several advanced model reduction techniques [1–5], 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. Modal reduction produces transfer function modal equivalents from the knowledge of the dominant poles and their corresponding residues, but requires specialized eigensolution methods that are still not capable enough to produce automatically the full set of truly dominant poles [6–8] for large system models. A good survey on model reduction methods employing either singular value decompositions or moment matching based methods is found in [9, 10]. A good introduction on modal model reduction on state space models can be found in [11], but a review on reduction methods that has applications extended to large, sparse descriptor system models is not known to the authors.

In this article, a new variant of the Dominant Pole Algorithm (DPA) [6] and the Dominant Pole Spectrum Eigensolver (DPSE) [7] 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. The SADPA directly operates on implicit state space systems, also known as descriptor systems, which are very sparse in practical power system applications.

The article is organized as follows. Section 2 summarizes some well known properties of scalar transfer functions and formulates the problem of computing the dominant poles of a scalar transfer function. Section 3 discusses the DPSE algorithm [7]. In section 4, the new variant SADPA is described. Extensive numerical results are presented in 5. Section 6 concludes.

*Joost Rommes is with Mathematical Institute, Utrecht University, P.O.Box 80100, 3508 TA Utrecht, The Netherlands (rommes@math.uu.nl).

†Nelson Martins is with CEPEL, P.O.Box 68007 - Rio de Janeiro, RJ - 20001 - 970, Brazil (nelson@cepel.br).

2 Transfer functions and dominant poles

The transfer function of a Single Input Single Output (SISO) system

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

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

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

where $I \in \mathbb{R}^{n\times n}$ is the identity matrix and $s \in \mathbb{C}$. Without loss of generality, $d = 0$ in the following.

Let the eigenvalues (poles) of A and the corresponding right and left eigenvectors be given by the triplets $(\lambda_j, \mathbf{x}_j, \mathbf{v}_j)$, and let the right and left eigenvectors be scaled so that $\mathbf{v}_j^*\mathbf{x}_j = 1$. Note that $\mathbf{v}_j^*\mathbf{x}_k = 0$ for $j \neq k$. The transfer function $H(s)$ can be expressed as a sum of residues R_j over first order poles [12]:

$$H(s) = \sum_{j=1}^{n} \frac{R_j}{s - \lambda_j},$$

where the residues R_j are

$$R_j = (\mathbf{x}_j^T\mathbf{c})(\mathbf{v}_j^*\mathbf{b}).$$

A pole λ_j that corresponds to a residue R_j with large magnitude $|R_j|$ is called a dominant pole, i.e. a pole that is well observable and controllable in the transfer function. This can also be observed from the corresponding Bode Magnitude plot of $H(s)$, where peaks occur at frequencies close to the imaginary parts of the dominant poles of $H(s)$. An approximation of $H(s)$ that consists of $k < n$ terms with $|R_j|$ above some value, determines the effective transfer function behaviour [13]:

$$H_k(s) = \sum_{j=1}^{k} \frac{R_j}{s - \lambda_j},$$

The problem of concern can now be formulated as: Given a SISO linear, time invariant, dynamical system $(A, \mathbf{b}, \mathbf{c}, d)$, compute $k \ll n$ dominant poles λ_j and the corresponding right and left eigenvectors $\mathbf{x}_j$ and $\mathbf{v}_j$.

3 Dominant Pole Spectrum Eigensolver

The DPSE [7] is based on the Dominant Pole Algorithm (DPA) [6] and Refactored Bi-Iteration (RBI) [14]. A scalar transfer function (1) is equal to its transpose:

$$H(s) = \frac{Y(s)}{U(s)} = \mathbf{c}^T(sI - A)^{-1}\mathbf{b} = \mathbf{b}^T(sI - A)^{-T}\mathbf{c}, \tag{2}$$

where $Y(s)$ and $U(s)$ are the Laplace transforms of $y(t)$ and $u(t)$. In matrix form, (2) becomes

$$\begin{bmatrix} sI - A & -\mathbf{b} \\ \mathbf{c}^T & 0 \end{bmatrix} \begin{bmatrix} X(s) \\ U(s) \end{bmatrix} = \begin{bmatrix} sI - A^T & \mathbf{c} \\ -\mathbf{b}^T & 0 \end{bmatrix} \begin{bmatrix} V(s) \\ U(s) \end{bmatrix} = \begin{bmatrix} 0 \\ Y(s) \end{bmatrix}, \tag{3}$$

where $X(s)$ and $V(s)$ are the Laplace transforms of the state vectors for A and A^T. The idea behind DPA, that is also used in DPSE, is that a pole of H can be defined as a $\lambda \in \mathbb{C}$ for which $\lim_{s\to\lambda} H(s) = \infty$, and hence $U(s) \to 0$ for $s \to \lambda$ and finite $Y(s)(= 1)$. The vectors $X(s)$ and $V(s)$ in (3) converge to left and right eigenvectors of A. The DPA is summarized in algorithm 1. All algorithms are described as directly operating on the state-space model. The practical implementations operate on the sparse descriptor system model, which is the unreduced Jacobian for the power system stability problem, analized in the examples of this paper (see section 5).

Algorithm 1 The Dominant Pole Algorithm.

INPUT: Initial pole estimate s_1

OUTPUT: Dominant pole λ and corresponding right and left eigenvectors $\mathbf{x}$ and $\mathbf{v}$.

1: Set $k = 1$

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

3: Solve $\mathbf{x} = X(s_k) \in \mathbb{C}^n$ and $u = U(s_k) \in \mathbb{C}$ from

$$\begin{bmatrix} s_k I - A & -\mathbf{b} \\ \mathbf{c}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ u \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

4: Solve $\mathbf{v} = V(s_k) \in \mathbb{C}^n$ and $u = U(s_k) \in \mathbb{C}$ from

$$\begin{bmatrix} (s_k I - A)^* & \mathbf{c} \\ -\mathbf{b}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ u \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

5: Compute the new pole estimate

$$s_{k+1} = s_k + \frac{u}{\mathbf{v}^* \mathbf{x}}$$

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

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

for some $\epsilon \ll 1$

7: Set $k = k + 1$

8: **end while**

The DPA can be proven to be a Newton process [6, 15] and converges asymptotically quadratically. Note that other criteria for convergence can be used [6, 7]. Note also that the complex transpose, denoted by A^*, is used in all algorithms, instead of the transpose A^T.

In the DPA, there is one moving shift s_k. The RBI uses multiple moving shifts and this is the second ingredient for the DPSE. By using multiple moving shifts, more than one dominant pole can be computed. Computation of repeated poles is avoided by using different initial shifts. The DPSE is summarized in algorithm 2. In a practical implementation, the number of moving shifts is decreased as soon as a pole has converged, while the corresponding right and left eigenvectors are kept in the matrices V and X. Furthermore, if a complex pole has converged, its complex conjugate is also a pole and the matrices V and X can be expanded with the corresponding complex conjugate right and left eigenvectors (this leads to slightly larger interaction matrices G and T).

4 A new approach: Subspace Accelerated DPA

In this section, three strategies to improve the DPSE will be discussed and combined to produce the new algorithm SADPA. In fact, the new algorithm SADPA can be seen as a generalization of the DPA to compute more than one dominant pole. Firstly, subspace acceleration, a well known technique for iterative methods, will be described. Secondly, a new selection strategy will be used to select the most dominant pole approximation and corresponding right and left eigenvector approximation every iteration. Thirdly, deflation will be used to avoid convergence to eigentriplets that are already found.

4.1 Subspace acceleration

A drawback of the DPSE 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

Algorithm 2 The Dominant Pole Spectrum Eigensolver.

INPUT: m initial pole estimates $s_1^{(i)}$, $i = 1, \ldots, m$
OUTPUT: m dominant pole triplets $(\lambda_i, \mathbf{x}_i, \mathbf{v}_i)$, $i = 1, \ldots, m$

Set $k = 1$
while not all converged **do**
  for $i = 1, \ldots, m$ **do**
    Solve $\mathbf{x}^{(i)} = X^{(i)}(s_k) \in \mathbb{C}^n$ from

$$\begin{bmatrix} s_k^{(i)} I - A & -\mathbf{b} \\ \mathbf{c}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x}^{(i)} \\ u \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

    Solve $\mathbf{v}^{(i)} = V^{(i)}(s_k) \in \mathbb{C}^n$ from

$$\begin{bmatrix} (s_k^{(i)} I - A)^* & \mathbf{c} \\ -\mathbf{b}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v}^{(i)} \\ u \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

  end for
  Set $V = [\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(m)}]$ and $X = [\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}]$
  Compute $G = V^* X$ and $T = V^* A X$
  Compute the new pole estimates

$$s_{k+1}^{(i)} = \lambda_i(H, G), \quad i = 1, \ldots, m$$

  A pole $\lambda_i = s_{k+1}^{(i)}$ has converged if

$$||A\mathbf{x}^{(i)} - s_{k+1}^{(i)} \mathbf{x}^{(i)}||_2 < \epsilon$$

  for some $\epsilon \ll 1$
  Set $k = k + 1$
end while

estimates $s_{k+1}^{(i)}$. The subspaces V and X, however, contain also information about other dominant eigentriplets (i.e. components in the direction of the corresponding eigenvectors) and the idea is now to use this information as well. Reasoning this way leads to a generalization of the DPA.

A global overview of the SADPA is shown in algorithm 3. Starting with a single shift s_1, the first iteration is equivalent to the first iteration of the DPA, but instead of discarding the corresponding right and left eigenvector approximations $\mathbf{x}_1$ and $\mathbf{v}_1$, they are kept in spaces X and V. In the next iteration, these spaces are expanded orthogonally, by using modified Gram-Schmidt (MGS) [16], with the approximations $\mathbf{x}_2$ and $\mathbf{v}_2$ corresponding to the new shift s_2 (see section 4.3). Hence the spaces grow and will contain better approximations. This idea is known as subspace acceleration. Approximations of the dominant poles are computed like in the DPSE, but the next question is to select the most promising approximation as new shift: unlike the DPSE, only one shift per iteration is used.

Algorithm 3 Subspace Accelerated DPA.

INPUT: System $(A, \mathbf{b}, \mathbf{c})$, initial pole estimate s_1 and the number of wanted poles p_{max}
OUTPUT: Dominant pole triplets $(\lambda_i, \mathbf{r}_i, \mathbf{l}_i)$, $i = 1, \ldots, p_{max}$

$k = 1$, $p_{found} = 0$, $\Lambda = R = L = []$
while $p_{found} < p_{max}$ **do**
  Solve $\mathbf{x} = X(s_k) \in \mathbb{C}^n$ from

$$\begin{bmatrix} s_k I - A & -\mathbf{b} \\ \mathbf{c}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ u \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

  Solve $\mathbf{v} = V(s_k) \in \mathbb{C}^n$ from

$$\begin{bmatrix} (s_k I - A)^* & \mathbf{c} \\ -\mathbf{b}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ u \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

  $X = \text{Expand}(X, R, L, \mathbf{x})$ {Alg. 5}
  $V = \text{Expand}(V, L, R, \mathbf{v})$ {Alg. 5}
  Compute $G = V^* X$ and $T = V^* A X$
  Compute eigentriplets of (T, G):

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

  Compute approximate eigentriplets of A as

$$(\hat{\lambda}_i = \tilde{\lambda}_i, \hat{\mathbf{x}}_i = X\tilde{\mathbf{x}}_i, \hat{\mathbf{v}}_i = V\tilde{\mathbf{v}}_i), \quad i = 1, \ldots, k$$

  $\widehat{\Lambda} = [\hat{\lambda}_1, \ldots, \hat{\lambda}_k]$
  $\widehat{X} = [\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_k]$
  $\widehat{V} = [\hat{\mathbf{v}}_1, \ldots, \hat{\mathbf{v}}_k]$
  $(\widehat{\Lambda}, \widehat{X}, \widehat{V}) = \text{Sort}(\widehat{\Lambda}, \widehat{X}, \widehat{V}, \mathbf{b}, \mathbf{c})$ {Alg. 4}
  if $||A\hat{\mathbf{x}}_1 - \hat{\lambda}_1 \hat{\mathbf{x}}_1||_2 < \epsilon$ **then**
    $(\Lambda, R, L, X, V) =$
      $\text{Deflate}(\hat{\lambda}_1, \hat{\mathbf{x}}_1, \hat{\mathbf{v}}_1, \Lambda, R, L, \widehat{X}_{2:k}, \widehat{V}_{2:k})$ {Alg. 6}
    $p_{found} = p_{found} + 1$
    Set $\hat{\lambda}_1 = \hat{\lambda}_2$
  end if
  Set $k = k + 1$
  Set the new pole estimate $s_{k+1} = \hat{\lambda}_1$
end while

4.2 Selection strategy

In algorithm 3, one has to choose a strategy to select the new pole estimate s_k. A possible choice is to use the generalized Rayleigh quotient as it is used in DPA and DPSE. Here, however, also another choice is possible, that is closer to the goal of computing the dominant poles.

Because in iteration k the interaction matrices $G \in \mathbb{C}^{k\times k}$ and $T \in \mathbb{C}^{k\times k}$ are of low order $k \ll n$ (see step 7 in Alg. 3), it is relatively cheap to compute the full eigendecomposition of the pencil (T, G). This provides k approximate eigentriplets $(\hat{\lambda}_i, \hat{\mathbf{x}}_i, \hat{\mathbf{v}}_i)$. The most natural thing to do is to choose the triplet $(\hat{\lambda}_j, \hat{\mathbf{x}}_j, \hat{\mathbf{v}}_j)$ with the most dominant pole approximation: compute the corresponding residues $\widehat{R}_i = (\hat{\mathbf{x}}_i^T \mathbf{c})(\hat{\mathbf{v}}_i^* \mathbf{b})$ of the k pairs and select the pole with the largest $|\widehat{R}_i|$ (see Alg. 4). The SADPA then continues with the new shift $s_{k+1} = \hat{\lambda}_j$.

Algorithm 4 $(\Lambda, X, V) = \mathrm{Sort}(\Lambda, X, V, \mathbf{b}, \mathbf{c})$

INPUT: $\Lambda \in \mathbb{C}^n$, $X, V \in \mathbb{C}^{n\times k}$, $\mathbf{b}, \mathbf{c} \in \mathbb{C}^n$
OUTPUT: $\Lambda \in \mathbb{C}^n$, $X, V \in \mathbb{C}^{n\times k}$ with λ_1 the pole with largest residue magnitude and $\mathbf{x}_1$ and $\mathbf{v}_1$ the corresponding approximate right and left eigenvectors.
1: Compute residues $R_i = (\mathbf{x}_i^T \mathbf{c})(\mathbf{v}_i^* \mathbf{b})$
2: Sort Λ, X, V in decreasing $|R_i|$ order

4.3 Deflation

Every iteration a convergence test is done like in DPA and DPSE: if for the selected eigentriplet $(\hat{\lambda}_j, \hat{\mathbf{x}}_j, \hat{\mathbf{v}}_j)$ the norm of the residual $||A\hat{\mathbf{x}}_j - \hat{\lambda}_j \hat{\mathbf{x}}_j||_2$ 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. A well known technique to avoid repeated computation is to use deflation [17].

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

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

has the same eigentriplets as A, but with the found eigenvalues transformed to zero.

Using this, the space X needs to be orthogonally expanded with $\Pi_j (I - \frac{\mathbf{x}_j \mathbf{v}_j^*}{\mathbf{v}_j^* \mathbf{x}_j}) \cdot \mathbf{x}$ and similarly, the space V needs to orthogonally expanded with $\Pi_j (I - \frac{\mathbf{v}_j \mathbf{x}_j^*}{\mathbf{x}_j^* \mathbf{v}_j}) \cdot \mathbf{v}$. These projections are implemented using modified Gram-Schmidt (MGS) (see Alg. 5).

Algorithm 5 $X = \mathrm{Expand}(X, R, L, \mathbf{x})$

INPUT: $X \in \mathbb{C}^{n\times k}$ with $X^*X = I$, $R, L \in \mathbb{C}^{n\times p}$, $\mathbf{x} \in \mathbb{C}^n$
OUTPUT: $X \in \mathbb{C}^{n\times(k+1)}$ with $X^*X = I$ and $\mathbf{x}_{k+1} = \Pi_{j=1}^p (I - \frac{\mathbf{r}_j \mathbf{l}_j^*}{\mathbf{l}_j^* \mathbf{r}_j}) \cdot \mathbf{x}$
1: $\mathbf{x} = \Pi_{j=1}^p (I - \frac{\mathbf{r}_j \mathbf{l}_j^*}{\mathbf{l}_j^* \mathbf{r}_j}) \cdot \mathbf{x}$
2: $\mathbf{x} = \mathrm{MGS}(X, \mathbf{x})$
3: $X = [X, \mathbf{x}/||\mathbf{x}||_2]$

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 conjugated pair is counted as one pole. The complete deflation procedure is shown in algorithm 6.

Algorithm 6
$(\Lambda, R, L, \widetilde{X}, \widetilde{V}) = \text{Deflate}(\lambda, \mathbf{x}, \mathbf{v}, \Lambda, R, L, X, V)$

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

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

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

4.4 Further improvements and remarks

It may happen that the spaces X and V become large, especially if a large number of dominant poles is wanted. A common way to deal with this is to do an implicit restart [17]: if the spaces X and V reach a certain maximum dimension $k_{max} \ll n$, they are reduced to a dimension $k_{min} < k_{max}$ by keeping the k_{min} most dominant approximate eigentriplets; the process is restarted with the reduced X and V. This procedure is repeated until all poles are found.

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

The SADPA requires only one initial shift, while the DPSE requires m initial shifts if m dominant poles are wanted. If rather accurate initial estimates are available, one can take advantage of this in SADPA as well by setting the next shift after deflation to a new initial estimate (step 20 of Alg. 3).

Every iteration, two systems of size $n+1$ need to be solved (step 3 and 4). As is also mentioned in [7], this can be efficiently done by computing one LU-factorization and solving the systems by using L and U, and U^* and L^* respectively. Because in practice the sparse Jacobian is used, computation of the LU-factorization is inexpensive.

The selection strategy can easily be changed to use another of the several existing indices of modal dominance [11, 18]. For instance, a good strategy for selecting dominant poles is: select the pole λ_i with largest $|R_i|/|\text{Re}(\lambda_i)|$ for a complex estimate, and the largest $|R_i|/|\lambda_i|$ for a real estimate. Futhermore, the strategy can be restricted to consider only poles in a certain frequency range. Also, instead of providing the number of wanted poles, the procedure can be automated even further by providing the desired maximum error $|H(s) - H_k(s)|$ for a certain frequency range: the procedure continues computing new poles until the error bound is reached. Note that such an error bound requires that the transfer function of the complete model can be computed efficiently

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

Transfer function	#poles	#LU	Time (s)
$W_{5061}/Vref_{5061}$	60	459	358
$W_{6405}/Vref_{6405}$	60	466	366
$W_{1155}/Vref_{1155}$	60	528	430
$W_{1107}/Vref_{1107}$	60	646	492
P_{sc}/B_{sc}	30	341	200
$Pt_{501}/Pref_{501}$	80	533	450

(which is usually the case for sparse descriptor systems).

5 Numerical Results

The algorithm 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. 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 about 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 ($\lambda_{ns} = -0.034 + j1.018$).

In the experiments, the convergence tolerance used was $\epsilon = 10^{-10}$. The spaces X and V were limited to dimension $n \times 10$ (i.e. a restart, with $k_{min} = 1$, 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. The DPSE uses deflation of complex conjugate pairs. All experiments are carried out in Matlab 6.5 [19] on an Intel Centrino Pentium 1.5 GHz with 512 MB RAM.

The state space realization of the BIPS model has 1664 states. The sparse, unreduced Jacobian has dimension 13251. Like the experiments in [6, 7], 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 (60, 30 or 80) 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 1 and the Bode plots of the corresponding modal equivalents, complete models and errors for the first four transfer functions are in Fig. 1 to Fig. 4. It is clearly observable that the reduced models capture the important dynamics. For completeness, the 15 most dominant poles and corresponding residues of $W_{6405}/Vref_{6405}$ are shown in Table 2. Note that SADPA succeeds in finding both real and complex poles.

The DPSE has more difficulties to converge as the number of shifts increases: typically, the tolerance is not reached for an increasing number of poles. Besides that, also the computational costs increase rapidly as the number of shifts increases, because the interaction matrices grow. To compute 60 poles with the DPSE, it must 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. The 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 strategy prevents repeated computation of the same pole. Furthermore,

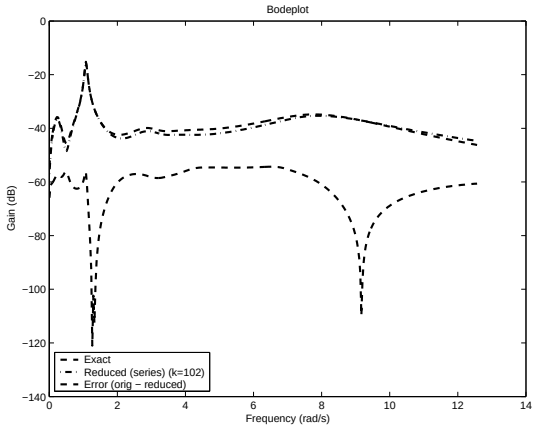

Figure 1: Bode plot of modal equivalent, complete model and error for $W_{5061}/Vref_{5061}$ (102 states).

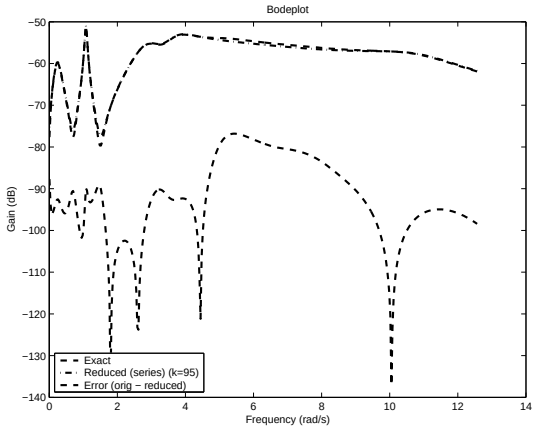

Figure 2: Bode plot of modal equivalent, complete model and error for $W_{6405}/Vref_{6405}$ (95 states).

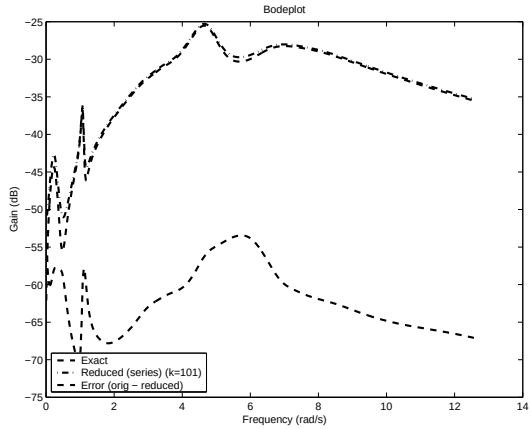

Figure 3: Bode plot of modal equivalent, complete model and error for $W_{1155}/Vref_{1155}$ (101 states).

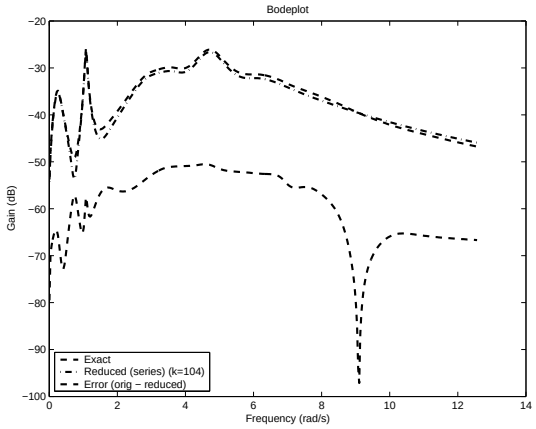

Figure 4: Bode plot of modal equivalent, complete model and error for $W_{1107}/Vref_{1107}$ (104 states).

Table 2: The 15 most dominant poles and corresponding residues of $W_{6405}/Vref_{6405}$.

Num.	Modes		Residues	
Mode	Real	Imaginary	Magnitude	Phase
1	-20.539	$\pm 1.0930j$	$6.2238 \cdot 10^{-3}$	± 65.000
3	-24.052		$5.4967 \cdot 10^{-3}$	± 180.00
4	-6.4446	$\pm 0.0715j$	$5.1620 \cdot 10^{-3}$	± 144.03
6	-10.068	$\pm 1.1975j$	$3.8408 \cdot 10^{-3}$	± 17.568
8	-7.5118	$\pm 0.2321j$	$3.6249 \cdot 10^{-3}$	± 128.70
10	-4.9203		$3.3856 \cdot 10^{-3}$	0
11	-2.9445	$\pm 4.8214j$	$3.0218 \cdot 10^{-3}$	± 167.72
13	-5.8881	$\pm 0.5484j$	$1.0877 \cdot 10^{-3}$	± 2.9382
15	-18.975		$1.0659 \cdot 10^{-3}$	0
16	-2.3488	$\pm 0.1100j$	$1.0138 \cdot 10^{-3}$	± 4.1653
18	-4.0233	$\pm 4.2124j$	$1.0087 \cdot 10^{-3}$	± 35.777
20	-6.4231	$\pm 8.6949j$	$9.9281 \cdot 10^{-4}$	± 97.729
22	-18.989		$9.8502 \cdot 10^{-4}$	0
23	-2.8052	$\pm 11.551j$	$7.5584 \cdot 10^{-4}$	± 8.3854
25	-0.5208	$\pm 2.8814j$	$7.5417 \cdot 10^{-4}$	± 139.68

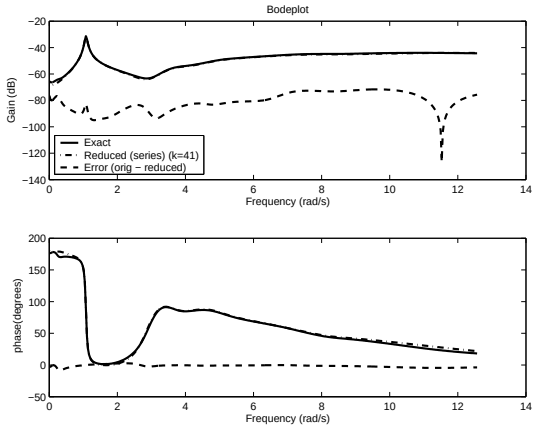

Figure 5: Bode plot of modal equivalent, complete model and error for the North-South mode damping of the Brazilian system ($P_{sc}(s)/B_{sc}(s)$) having 46 PSSs (1664 states in the complete model, 41 in the reduced model).

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. In the neighborhood of exact (dominant) poles of the transfer function, SADPA has quadratic convergence [7].

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 [20–22]. 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 [20]. Fig. 5 shows the frequency response of the complete model and the reduced model (41 states) together with the error. Fig. 6 shows the corresponding step response (step 0.01). 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 (30 poles, 56 states) was computed by SADPA in 200 seconds (341 factorizations). This reduced model was reduced to a 41 state (22 poles) by removing less dominant contributions. A table with the dominant poles and corresponding residues can be found in [20].

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 appplied to its speed-governor reference. For this transfer function it is known from numerical experiments that the DPSE has difficulties in producing a good modal equivalent: several DPSE attempts are needed to obtain an acceptable modal equivalent. The SADPA is able to *automatically* produce good results for both the frequency and step response, as can be observed from Fig. 7 and Fig. 8. The SADPA computed the reduced model (80 poles, 118 states) in 450 seconds (533 factorizations) using just a single initial shift $s_1 = 1i$, in one single run. For this example, the selection strategy was set to selecting the pole λ_i with largest $|R_i|/|\mathrm{Re}(\lambda_i)|$ every iteration.

It must be noted that all modal equivalents in this section can be reduced even further by neglecting less dominant contributions, or by application of the Balanced Model Reduction algorithm [11] to a state space realization of the modal equivalent, as described in a paper submitted to Trans. on Power Systems [20].

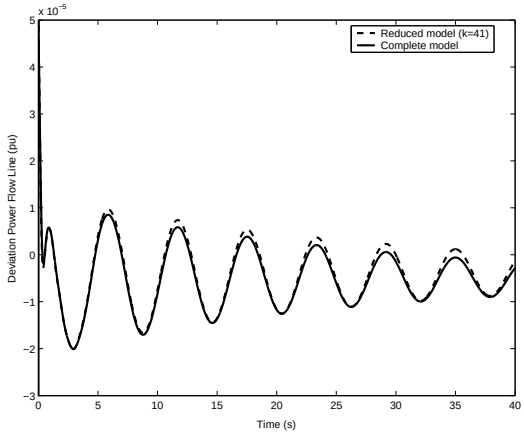

Figure 6: Open-loop (no POD controllers) step response (step 0.01) of modal equivalent and complete model for the North-South mode damping of the Brazilian system ($P_{sc}(s)/B_{sc}(s)$) having 46 PSSs (1664 states in the complete model, 41 in the reduced model).

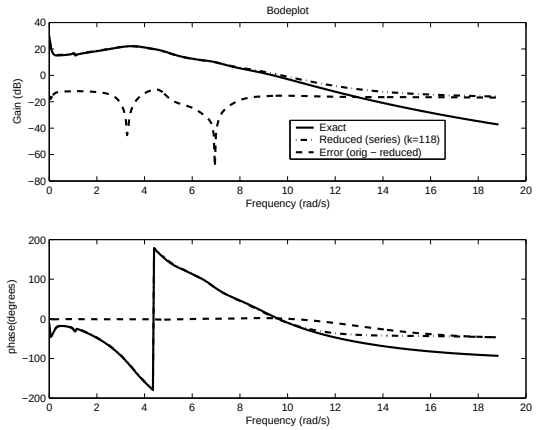

Figure 7: Bode plot of modal equivalent, complete model and error for the North-South Brazilian system transfer function $Pt_{501}(s)/Pref_{501}(s)$ (1664 states in the complete model, 118 in the reduced model).

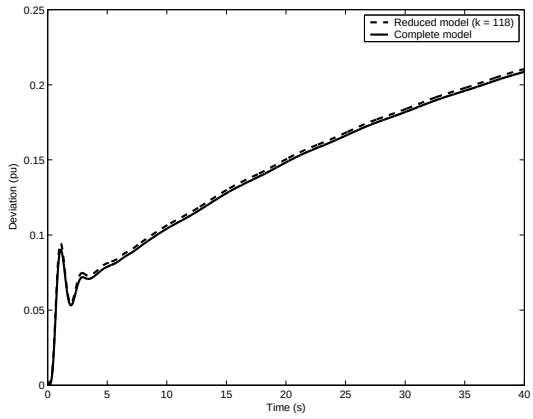

Figure 8: Step response (step 0.01) of modal equivalent and complete model for the North-South Brazilian system transfer function $Pt_{501}(s)/Pref_{501}(s)$ (1664 states in the complete model, 118 in the reduced model).

6 Conclusions

The algorithm described in this paper, SADPA, is a fast method to compute the dominant poles and corresponding eigenvectors of a scalar transfer function. It has several advantages compared to 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 deflation techniques. Fourthly, SADPA is completely automatic: with a single shift it is able to compute as many dominant poles as wanted, without intermediate human interaction.

The algorithm as presented in this article should be adequate for computer implementation by an experienced programmer. The paper results are related to the analysis and control of small signal stability, but the SADPA algorithm is general and could be effectively applied to problems in other engineering fields that allow sparse descriptor system formulations.

Currently, a generalization of the algorithm to MIMO systems is being developed [8].

Acknowledgment

The first author thanks Gerard Sleijpen for pointing to the work of Nelson Martins. Both authors thank CEPEL and ELETROBRAS for providing the test systems.

References

[1] J. J. Sanchez-Gasca and J. H. Chow, "Power System Reduction to Simplify the Design of Damping Controllers for Interarea Oscillations," *IEEE Trans. Power Syst.*, vol. 11, no. 3, pp. 1342–1349, Aug 1996.

[2] B. C. Pal, A. H. Cooninck, and B. J. Cory, "Robust Damping of Inter-area Oscillations in Power Systems with Superconducting Magnetic Energy Storage Devices," *Proc. IEE Generation, Transmission and Distribution*, vol. 146, pp. 633–639, Nov 1999.

[3] D. Chaniotis and M. A. Pai, "Model Reduction in Power Systems using Krylov Subspace Methods," *IEEE Trans. Power Syst.*, vol. 20, no. 2, pp. 888–894, May 2005.

[4] A. Ramirez, A. Semlyen, and R. Iravani, "Order Reduction of the Dynamic Model of a Linear Weakly Periodic System-Part i: General Methodology," *IEEE Trans. Power Syst.*, vol. 19, no. 2, pp. 857–865, May 2004.

[5] G. Troullinos, J. Dorsey, H. Wong, and J. Myers, "Reducing the Order of Very Large Power System Models," *IEEE Trans. Power Syst.*, vol. 3, no. 1, pp. 127–133, Feb 1988.

[6] N. Martins, L. T. G. Lima, and H. J. C. P. Pinto, "Computing Dominant Poles of Power System Transfer Functions," *IEEE Trans. Power Syst.*, vol. 11, no. 1, pp. 162–170, Feb 1996.

[7] N. Martins, "The Dominant Pole Spectrum Eigensolver," *IEEE Trans. Power Syst.*, vol. 12, no. 1, pp. 245–254, Feb 1997.

[8] N. Martins and P. E. M. Quintão, "Computing Dominant Poles of Power System Multivariable Transfer Functions," *IEEE Trans. Power Syst.*, vol. 18, no. 1, pp. 152–159, Feb 2003.

[9] A. C. Antoulas and D. C. Sorensen, "Approximation of large-scale dynamical systems: an overview," *Int. J. Appl. Math. Comput. Sci.*, vol. 11, no. 5, pp. 1093–1121, 2001.

[10] A. C. Antoulas, *Approximation of Large-Scale Dynamical Systems.* SIAM, 2005.

[11] M. Green and D. J. N. Limebeer, *Linear Robust Control.* Prentice-Hall, 1995.

[12] T. Kailath, *Linear Systems*. Prentice-Hall, 1980.

[13] J. R. Smith, J. F. Hauer, D. J. Trudnowski, F. Fatehi, and C. S. Woods, "Transfer Function Identification in Power System Application," *IEEE Trans. Power Syst.*, vol. 8, no. 3, pp. 1282–1290, Aug 1993.

[14] J. M. Campagnolo, N. Martins, and D. M. Falcão, "Refactored Bi-Iteration: A High Performance Eigensolution Method for Large Power Systems," *IEEE Trans. Power Syst.*, vol. 11, no. 3, pp. 1228–1235, Aug 1996.

[15] L. H. Bezerra, "Written discussion to [6]," *IEEE Trans. Power Syst.*, vol. 11, no. 1, p. 168, Feb 1996.

[16] G. H. Golub and C. F. van Loan, *Matrix Computations*, 3rd ed. John Hopkins University Press, 1996.

[17] Y. Saad, *Numerical methods for large eigenvalue problems: theory and algorithms*. Manchester University Press, 1992.

[18] L. A. Aguirre, "Quantitative Measure of Modal Dominance for Continuous Systems," in *Proc. of the 32nd Conference on Decision and Control*, December 1993, pp. 2405–2410.

[19] The Mathworks, Inc., "Matlab R13." [Online]. Available: http://www.mathworks.com

[20] N. Martins, F. G. Silva, and P. C. Pellanda, "Utilizing Transfer Function Modal Equivalents of Low-order for the Design of Power Oscillation Damping Controllers in Large Power Systems," in *IEEE/PES General Meeting*, June 2005, pp. 2642–2648, enlarged version submitted to IEEE Trans. Power Syst.

[21] C. Gama, L. Ängquist, G. Ingeström, and M. Noroozian, "Commissioning and operative experience of tcsc for damping power oscillation in the brazilian north-south interconnection," in *Proc. CIGRÉ Session 2000*, 2000, paper 14-104.

[22] CIGRE TF 38.02.16, "Impact of the interactions among power system controls," CIGRE, Tech. Rep. 166, July 2000.