LARGE-SCALE ESTIMATION OF DOMINANT POLES OF A TRANSFER FUNCTION BY AN INTERPOLATORY FRAMEWORK

EMRE MENGI*

Abstract. We focus on the dominant poles of the transfer function of a descriptor system. The transfer function typically exhibits large norm at and near the imaginary parts of the dominant poles. Consequently, the dominant poles provide information about the points on the imaginary axis where the $\mathcal{L}_\infty$ norm of the system is attained, and they are also sometimes useful to obtain crude reduced-order models. For a large-scale descriptor system, we introduce a subspace framework to estimate a prescribed number of dominant poles. At every iteration, the large-scale system is projected into a small system, whose dominant poles can be computed at ease. Then the projection spaces are expanded so that the projected system after subspace expansion Hermite interpolates the large-scale system at the computed dominant poles. We prove an at-least-quadratic-convergence result for the framework, and provide numerical results confirming this. On real benchmark examples, the proposed framework appears to be more reliable than SAMDP [IEEE Trans. Power Syst. 21, 1471-1483, 2006], one of the widely used algorithms due to Rommes and Martins for the estimation of the dominant poles, for which we provide a theoretical explanation.

Key words. dominant pole, descriptor system, large scale, projection, Hermite interpolation, model order reduction

AMS subject classifications. 65F15, 93C05, 93A15, 34K17

1. Introduction.

The dominant poles of a descriptor system provide substantial insight into the behavior of the transfer function of the system. Here, we consider a descriptor system with the state-space representation

$$Ex'(t) = Ax(t) + Bu(t), \qquad y(t) = Cx(t) + Du(t), \tag{1.1}$$

and the transfer function

$$H(s) := C(sE - A)^{-1}B + D, \tag{1.2}$$

where $A, E \in \mathbb{C}^{n\times n}$, $B \in \mathbb{C}^{n\times m}$, $C \in \mathbb{C}^{p\times n}$, $D \in \mathbb{C}^{p\times m}$ with $n \geq m,\ n \geq p$. This text deals with the estimation of a few dominant poles of a descriptor system of the form (1.1) in the large-scale setting when the order of the system n is large.

We assume throughout that $L(s) = A - sE$ is a regular pencil, i.e., $\det L(s)$ is not identically equal to zero at all s, and that its finite eigenvalues are simple. If E is singular, it is also assumed that the descriptor system has index one, that is 0 as an eigenvalue of $L_{\mathsf{pal}}(s) = E - sA$ is semi-simple. Many of the ideas in the subsequent discussions can possibly be generalized even if some of the finite eigenvalues of L are semi-simple, but not necessarily simple, excluding the discussions where we analyze the proposed framework. The index-one assumption is essential, as our interest in the dominant poles stems partly from estimating $\mathcal{L}_\infty$ norm, which is usually even not bounded for a system with index greater than one.

It follows from the Kronecker Canonical form [14] that there exist invertible matrices $W, V \in \mathbb{C}^{n\times n}$ such that

$$W^*AV = \begin{bmatrix} \Lambda & 0 \\ 0 & I_{n-\widetilde{n}} \end{bmatrix} =: \Lambda_A \quad \text{and} \quad W^*EV = \begin{bmatrix} I_{\widetilde{n}} & 0 \\ 0 & 0 \end{bmatrix} =: \Lambda_E, \tag{1.3}$$

*Koç University, Department of Mathematics, Rumeli Feneri Yolu 34450, Sarıyer, Istanbul, Turkey, E-Mail: emengi@ku.edu.tr.

where $\Lambda \in \mathbb{C}^{\widetilde{n}\times\widetilde{n}}$ is the diagonal matrix with the finite eigenvalues $\lambda_1, \ldots, \lambda_{\widetilde{n}}$ of L along its diagonal. Indeed, W and V can be expressed as

$$W = \begin{bmatrix} w_1 & \ldots & w_{\widetilde{n}} & W_\infty \end{bmatrix} \quad \text{and} \quad V = \begin{bmatrix} v_1 & \ldots & v_{\widetilde{n}} & V_\infty \end{bmatrix},$$

where v_j and w_j are right and left eigenvectors corresponding to λ_j such that $w_j^* E v_j = 1$ for $j = 1, \ldots, \widetilde{n}$, and $W_\infty, V_\infty \in \mathbb{C}^{n\times(n-\widetilde{n})}$ have linearly independent columns.

Using the Kronecker canonical form, transfer function (1.2) can be rewritten as

$$\begin{aligned} H(s) &= C(sW^{-*}\Lambda_E V^{-1} - W^{-*}\Lambda_A V^{-1})^{-1}B + D \\ &= (CV)(s\Lambda_E - \Lambda_A)^{-1}(W^*B) + D \\ &= \sum_{j=1}^{\widetilde{n}} \frac{(Cv_j)(w_j^*B)}{s-\lambda_j} + M_\infty + D, \end{aligned} \tag{1.4}$$

where $M_\infty := -(CV_\infty)(W_\infty^* B) \in \mathbb{C}^{n\times n}$ is constant (i.e., independent of s) and due to the infinite eigenvalues of L.

The poles of the system are the finite eigenvalues of L. The dominant ones among them are those that can cause large frequency response, i.e., the eigenvalues responsible for larger $\|H(\mathrm{i}\omega)\|_2$ at some $\mathrm{i}\omega$ on the imaginary axis. The formal definition is given below.

Definition 1.1 (Dominant Poles). *Let us order the finite eigenvalues $\lambda_1, \ldots, \lambda_{\widetilde{n}}$ of L as $\lambda_{i_1}, \ldots, \lambda_{i_{\widetilde{n}}}$ so that*

$$\frac{\|Cv_{i_1}\|_2\|w_{i_1}^*B\|_2}{|\mathrm{Re}\,\lambda_{i_1}|} \;\geq\; \frac{\|Cv_{i_2}\|_2\|w_{i_2}^*B\|_2}{|\mathrm{Re}\,\lambda_{i_2}|} \;\geq\; \ldots \;\geq\; \frac{\|Cv_{i_{\widetilde{n}}}\|_2\|w_{i_{\widetilde{n}}}^*B\|_2}{|\mathrm{Re}\,\lambda_{i_{\widetilde{n}}}|}. \tag{1.5}$$

The eigenvalue λ_{i_j} is called the jth dominant pole of the system in (1.1). We refer to the first dominant pole as simply the dominant pole.

There are other definitions of dominant poles employed in the literature. For instance, in [4] and [30] the dominant poles are defined based on the orderings of the finite eigenvalues according to $1/|\mathrm{Re}\,\lambda_j|$ and $\|Cv_j\|_2\|w_j^*B\|_2$ for $j = 1, \ldots, \widetilde{n}$, respectively. It should however be noted that Definition 1.1 for the dominant poles that we rely on throughout this text is the one that is most widely used in the literature. This definition also appears to be the meaningful one for instance for model order reduction and for the estimation of $\omega \in \mathbb{R}$ where $\|H(\mathrm{i}\omega)\|_2$ exhibits large peaks.

1.1. Motivation. A reduced order model can be obtained for (1.1) based on the transfer function

$$H_{\mathsf{red}}(s) \;=\; \sum_{j=1}^{r} \frac{(Cv_{i_j})(w_{i_j}^*B)}{s-\lambda_{i_j}} + M_\infty + D \tag{1.6}$$

for a prescribed positive integer $r < \widetilde{n}$. Indeed, assuming (1.1) is asymptotically stable, the $\mathcal{H}_\infty$-norm error for this reduced order model is

$$\|H - H_{\mathsf{red}}\|_{\mathcal{H}_\infty} \;=\; \sup_{\omega\in\mathbb{R}} \left\| \sum_{j=r+1}^{\widetilde{n}} \frac{(Cv_{i_j})(w_{i_j}^*B)}{\mathrm{i}\omega - \lambda_{i_j}} \right\|_2 \;\leq\; \sum_{j=r+1}^{\widetilde{n}} \frac{\|Cv_{i_j}\|_2\|w_{i_j}^*B\|_2}{\big|\mathrm{Re}\,\lambda_{i_j}\big|}.$$

This is known as modal model reduction. As noted in [4, Section 9.2], the convergence with respect to the order of the reduced model may be slow. Still, if a few

dominant poles can be estimated at ease, this approach provides a crude low order approximation.

But our motivation for the estimation of the dominant poles is mainly driven from the computation of $\mathcal{L}_\infty$ norm of a descriptor system. The $\mathcal{L}_\infty$ norm for (1.1) is defined by

$$\|H\|_{\mathcal{L}_\infty} := \sup_{\omega\in\mathbb{R}} \|H(\mathrm{i}\omega)\|_2 = \sup_{\omega\in\mathbb{R}} \sigma_{\max}(C(\mathrm{i}\omega E - A)^{-1}B + D), \tag{1.7}$$

where H is the transfer function in (1.2), and $\sigma_{\max}(\cdot)$ denotes the largest singular value of its matrix argument. If the system is asymptotically stable with poles on the open left-half in the complex plane, then the $\mathcal{L}_\infty$ norm is the same as the $\mathcal{H}_\infty$ norm of the system, which can be expressed as an optimization problem as in (1.7) but with the supremum over the right-half of the complex plane rather than over the imaginary axis. The $\mathcal{H}_\infty$ norm of the system is an indicator of robust stability, as indeed its reciprocal is equal to a structured stability radius of the system [20, 19]. Partly due to these robust stability considerations, if the system has design parameters, it is desirable to minimize the $\mathcal{H}_\infty$ norm of the system over the space of parameters; see e.g., [32, 2] and references therein. A related problem is the $\mathcal{H}_\infty$-norm model reduction problem [4], which concerns finding a nearest reduced order system with prescribed order with respect to the $\mathcal{H}_\infty$ norm. Such minimization tasks involving $\mathcal{H}_\infty$ norm in the objective may require a few $\mathcal{H}_\infty$-norm calculations. There are various approaches that are tailored for the estimation of the $\mathcal{L}_\infty$ norm of a large-scale descriptor system; see for instance [18, 9, 12, 25, 8]. However, the optimization problem in (1.7) is nonconvex, and these algorithms converge to local maximizers of the singular value function in (1.7), that are not necessarily optimal globally. This is for instance the case with our recent subspace framework [1] for large-scale $\mathcal{L}_\infty$-norm estimation, which starts with an initial reduced order model whose transfer function interpolates the transfer function (1.2) of the original large-scale system at prescribed points. If these locally convergent algorithms are started with good initial points, in the case of [1] good interpolation points, close to a global maximizer of the singular value function in (1.7), then convergence to this global maximizer occurs meaning that the $\mathcal{L}_\infty$ norm is computed accurately.

Good candidates for these initial points are provided by the imaginary parts of the dominant poles of (1.1), as global maximizers of the singular value function are typically close to the imaginary parts of dominant poles. This fact is illustrated in Figure 1.1, where on the left and on the right the imaginary parts of the most dominant five and seven poles of the systems are depicted on the horizontal axis with crosses. They are quite close to local maximizers where the singular value function exhibits highest peaks. Moreover, in both plots in Figure 1.1, one of these local maximizers is indeed a global maximizer.

We propose to use the imaginary parts of the dominant poles estimated by the approach introduced here for the initialization of the algorithms for large-scale $\mathcal{L}_\infty$-norm computation, e.g., as the initial interpolation points for the subspace framework of [1].

1.2. Literature and Our Approach. Several approaches have been proposed for the estimation of the dominant poles of a descriptor system in the literature. Most of these approaches stem from the dominant pole algorithm (DPA) [22], which is originally for the standard single-input-single-output LTI systems (i.e., for the descriptor systems as in (1.1) but with $E = I$ and $m = p = 1$), and which is a Newton iteration inspired from the Rayleigh-quotient iteration to find an eigenvalue of A. DPA

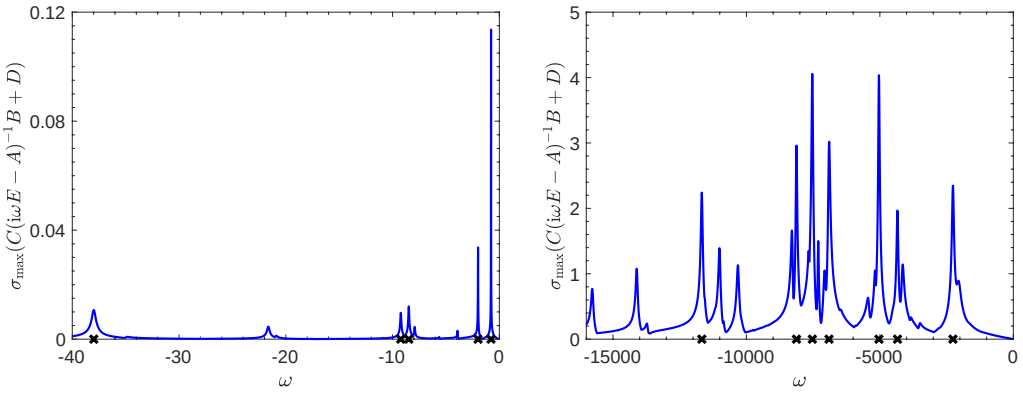


FIG. 1.1. *The plots of* $\sigma_{\max}(C(\mathrm{i}\omega E - A)^{-1}B + D)$ *as a function of* ω *for the* `iss` *system of order 270 (left) and* `M10PI_n` *system of order 625 (right) available in the SLICOT library. The black crosses mark the imaginary parts of the first five (left) and first seven (right) dominant poles of the systems.*

is meant to locate only one dominant pole. This is later generalized to locate several dominant poles in [21]. The generalized algorithm is referred as the dominant pole spectrum eigensolver (DPSE), and can be viewed as a simultaneous Rayleigh-quotient iteration. An extension that keeps all of the previous directions and performs subspace projections is described in [29]. The original DPA algorithm is later adapted for multiple-input-multiple-output systems in [23], and a variant that employs all of the previous directions for projection is introduced in [28]. The extensions of these algorithms for descriptor systems with an arbitrary E, possibly singular, is quite straightforward; see for instance the survey paper [27]. We also refer to [30] for a convergence analysis of DPA and its variants in the single-input-single-output case.

The subspace framework that we propose here differs from all of the existing methods for dominant pole estimation in two major ways. First, our approach performs projections explicitly on the state-space representation, similar to those employed by model reduction techniques. Secondly, our approach is built specifically for interpolation. At every iteration, it computes the dominant poles of a projected system. Then it expands projection subspaces so as to achieve certain Hermite interpolation properties between the original system and the projected system at these dominant poles of the projected system. The satisfaction of the Hermite interpolation properties is the reason for the quick convergence of the proposed framework at least at a quadratic rate under mild assumptions, which we prove in theory and illustrate in practice on real examples.

The Hermite interpolation between the original and projected system is achieved by means of extensions of classical Hermite interpolation results used by interpolatory model reduction techniques [13, 16, 7, 5, 17]. The interpolatory model reduction techniques are set up so as to form an interpolating reduced order system whose transfer function approximates the transfer function of the full system well. On the other hand, the approach here is tailored to form an interpolating reduced order system whose dominant poles approximate those of the full system well. Moreover, the interpolatory model reduction techniques enforce Hermite interpolation between the transfer functions of the full and reduced systems only in certain directions, whereas,

in the framework here for dominant pole estimation, the whole transfer function of the full system is Hermite interpolated by that of the reduced system at selected points in the complex plane.

The proposed framework is partly inspired from our previous work for $\mathcal{L}_\infty$-norm computation [1]. However, devising a rapidly converging subspace framework for dominant pole estimation and establishing its quick convergence come with additional challenges. By making an analogy with quasi-Newton methods, in [1] we maximize over the imaginary axis the largest singular value of the reduced transfer function, viewed as a model function, rather than the full transfer function. In the context here, there is no clear objective to be maximized. Instead, the quantities that we would like to compute, i.e., dominant poles, are hidden inside the transfer function. Thus, to explain the quick convergence and provide a formal rate-of-convergence argument, we devise a function (in particular f in (4.1)) over the complex plane whose minimizers are poles of $H(s)$, and view its reduced counterparts as model functions. The devised function and their reduced counterparts are tailored as analytic functions near the poles under mild assumptions; the rate-of-convergence analysis we present would not be applicable without such smoothness features. A second critical issue is that all the interpolation points and analysis in [1] are restricted to the imaginary axis, while, in the current work, all of the developments have to be over the whole complex plane. Tailoring a subspace framework over the complex plane requires care. For instance, it seems essential to interpolate at least the first two derivatives of the transfer functions in order to attain superlinear convergence, unlike the framework in [1] operating on the imaginary axis for which interpolating first derivatives suffices for superlinear convergence. The proposed framework has also similarities with those in [2] and [6] for the minimization of the $\mathcal{H}_\infty$ norm and for nonlinear eigenvalue problems. But again there are remarkable differences in the ways interpolation is performed, and in the rate-of-convergence analyses explaining quick convergence; to say the least, in those works the analytic functions on which the analyses operate on are readily available, and quite different than the one we set up here.

1.3. Contributions and Outline. We introduce the first subspace framework that makes use of rigorous and systematic use of interpolation for the estimation of the dominant poles of a large-scale descriptor system. The framework appears to be more reliable than the existing methods. On benchmark examples the framework proposed here typically returns dominant poles with larger dominance metrics compared to those dominant poles returned by the method in [28], commonly employed today for dominant pole estimation. We prove rigorously that the rate of convergence of the framework is at least quadratic with respect to the number of subspace iterations, which is also confirmed in practice on benchmark examples. The proposed framework is implemented rigorously, and this implementation is made publicly available. The interpolation result presented here for the transfer functions of descriptor systems (i.e., Lemma 3.1) generalizes the interpolation results in our previous works [1, Lemma 3.1], [6, Lemma 2.1].

Our presentation is organized as follows. The next section concerns an application of the dominant poles for estimating the stability radius of a linear dissipative Hamiltonian system, a problem closely connected to $\mathcal{L}_\infty$ norm. We describe the interpolatory subspace framework to compute a prescribed number of dominant poles in Section 3. In Section 4, the rate of convergence of the proposed subspace framework to compute the most dominant pole is analyzed. Section 5 is devoted to the practical details that have to be taken into consideration in an actual implementa-

tion of the framework such as initialization and termination criterion. The proposed framework is tested numerically in Section 6 on real benchmark examples used in the literature for dominant pole estimation and model order reduction. In this numerical experiments section, comparisons of the framework with the subspace accelerated multiple-input-multiple-output dominant pole algorithm [28] are reported. Section 7 provides a theoretical argument in support of the more robust convergence of the proposed framework to the dominant poles in comparison to the algorithm in [28].

2. Stability Radius of a Linear Dissipative Hamiltonian System. Computation of the $\mathcal{L}_\infty$-norm (1.7) for a large-scale system efficiently and with high accuracy is still not fully addressed today.

The level-set method due to Boyd and Balakrishnan [10], Bruinsma and Steinbuch [11] is extremely reliable and accurate. It is still the method to be used for a small-scale system. Unfortunately, it is not meant for large-scale systems, as, for a system of order n, it requires the calculation of all imaginary eigenvalues of $2n \times 2n$ matrices. The only other option is to use the locally convergent algorithms such as [18, 9, 12, 25, 1, 8]. These algorithms are better suited to cope with the large-scale setting, but, especially when the singular value function in (1.7) has many local maximizers, there is a decent chance that they will converge to a local maximizer that is not optimal globally. A particular problem that is connected to the computation of an $\mathcal{L}_\infty$ norm is the stability radius of a linear dissipative Hamiltonian (DH) system.

A linear DH system is a linear autonomous control system of the form

$$x'(t) \;=\; (J-R)Qx(t),$$

where $J, R, Q \in \mathbb{C}^{\widetilde{n}\times\widetilde{n}}$ are constant matrices such that $J^* = -J$, $R^* = R$, $Q^* = Q$, and R, Q are positive semidefinite, positive definite, respectively. Various applications in science and engineering give rise to linear DH systems [31, 15]. A linear DH system is always Lyapunov stable, that is all of the eigenvalues of $(J-R)Q$ are contained on the closed left-half plane and its eigenvalues on the imaginary axis (if there is any) are semi-simple. However, unstructured perturbations of J, R and/or Q may result in unstable systems with eigenvalues whose real parts are positive. In the presence of uncertainties on the matrix R, for given restriction matrices $B \in \mathbb{C}^{\widetilde{n}\times\widetilde{m}}$, $C \in \mathbb{C}^{\widetilde{p}\times\widetilde{n}}$ with $\widetilde{m} \le \widetilde{n}$, $\widetilde{p} \le \widetilde{n}$ on the perturbations of R due to uncertainties, the stability radius

$$r(R;B,C) := \inf\{\|\Delta\|_2 \mid \Delta \in \mathbb{C}^{\widetilde{m}\times\widetilde{p}},\ \Lambda\left((J-(R+B\Delta C))Q\right) \cap \mathrm{i}\mathbb{R} \neq \emptyset\}$$

is proposed in [24], where $\Lambda(\cdot)$ denotes the spectrum of its matrix argument, and $\mathrm{i}\mathbb{R}$ the set of purely imaginary numbers. It is also proven in [24] that $r(R;B,C)$ can be characterized as

$$r(R;B,C) \;=\; \frac{1}{\sup_{\omega\in\mathbb{R}} h(\omega)}, \qquad h(\omega) \;:=\; \sigma_{\max}(CQ(\mathrm{i}\omega I-(J-R)Q)^{-1}B) \tag{2.1}$$

provided that $r(R;B,C)$ is finite. Hence, $r(R;B,C)$ is the reciprocal of a special $\mathcal{L}_\infty$ norm as in (1.7) with CQ, $(J-R)Q$ taking place of C, A and $E = I$, $D = 0$.

The singular value function in (2.1) has usually many local maximizers especially when R has low rank. As argued in Section 1.1, a remedy to convergence to local maximizers that are not optimal globally is to use the dominant poles of the associated system

$$x'(t) \;=\; (J-R)Qx(t)+Bu(t), \qquad y(t) \;=\; CQx(t) \tag{2.2}$$

TABLE 2.1
Comparison of the subspaces frameworks [3] with dominant poles and [1, 3] with equally-spaced points on 3000 random linear DH systems in Section 2. Accuracy refers to the percentage of the results that differ from the results by the BB, BS algorithm by an amount less than 10^{-8}.

		# iterations		time in s	
method	accuracy	mean	median	mean	median
[1] with equally-spaced points	% 84.93	17.90	13	1.86	0.69
[3] with equally-spaced points	% 91.93	12.90	11	2.54	1.25
[3] with dominant poles	% 99.43	10.52	9	1.91	1.00
BB, BS Algorithm [10, 11]	% 100	—	—	6.61	6.39

for initialization. To illustrate this point, we compute $r(R; B, C)$ for 3000 random linear DH systems with $\widetilde{n} = 500$, $\widetilde{m} = \widetilde{p} = 2$[1] using the subspace framework [1] and the framework [3], the variant that respects the DH structure, both of which are locally convergent. For each one of these DH systems, the randomly-generated matrices J, R, Q, B, C are real, R is constrained to have rank in $[10, 50]$, and the singular value function $h(\omega)$ in (2.1) has typically at least thirty local maximizers and is even. As for the initial interpolation points, we consider the following two possibilities:

(equally-spaced points) 10 points among 40 equally-spaced points in $[-1200, 0]$ (which contains a global maximizer of $h(\omega)$) yielding the largest value of $h(\omega)$;

(dominant poles) 10 points among the imaginary parts of the 40 most dominant poles of (2.2) that yield the largest value of $h(\omega)$.

Four approaches are compared in Table 2.1. The BB, BS algorithm in the last line of the table refers to the level-set method due to Boyd and Balakrishnan [10], Bruinsma and Steinbuch [11]. Even the subspace frameworks [1, 3] benefit from this level-set method to solve the small projected problems. We remark that $n = 500$ is relatively small so that the BB, BS algorithm is applicable, in particular we can verify the correctness of the results computed by the frameworks by comparing them with the results returned by the BB, BS algorithm. It is apparent from the second column in Table 2.1 that [3] initialized with dominant poles is considerably more accurate than [1, 3] using equally-spaced points. The runtimes reported in the last column include the time for initialization, in particular the time for the computation of the dominant poles in the third line. Observe that [3] with dominant poles is substantially faster than direct applications of the level-set method [10, 11], and nearly as accurate as the level-set method.

3. The Proposed Subspace Framework. Some of the most widely-used approaches for model order reduction of descriptor systems perform Petrov-Galerkin projections. In the Petrov-Galerkin framework, given two subspaces $\mathcal{V}$, $\mathcal{W}$ and matrices V, W whose columns form orthonormal bases for these subspaces, system (1.1) is approximated by

$$W^*EVx'_{\mathsf{red}}(t) \;=\; W^*AVx_{\mathsf{red}}(t) + W^*Bu(t), \qquad y(t) \;=\; CVx_{\mathsf{red}}(t) + Du(t).$$

Note that this reduced order system is obtained from (1.1) by restricting the state space to $\mathcal{V}$ (i.e., by replacing $x(t)$ with $Vx_{\mathsf{red}}(t)$, which is merely an approximation)

[1]Such random linear DH systems can be generated using the Matlab routine `randomDH` made available on the web at https://zenodo.org/record/5103430, specifically with the command `randomDH(500,2,2)`.

and imposing the orthogonality of the resulting residual of the differential part to $\mathcal{W}$. Moreover, it is important that the dimensions of $\mathcal{V}$ and $\mathcal{W}$ are the same, so that W^*EV and W^*AV are square matrices, and poles of the reduced system are the eigenvalues of a square pencil, just like the original system.

Interpolation is a plausible strategy for the construction of the subspaces; $\mathcal{V}$, $\mathcal{W}$ can be formed so that the transfer function of the reduced system

$$H_{\mathsf{red}}^{\mathcal{W},\mathcal{V}}(s) := CV(sW^*EV - W^*AV)^{-1}W^*B + D \tag{3.1}$$

Hermite interpolates the transfer function $H(s)$ as in (1.2) of the original system at prescribed points. The following result indicates how Hermite interpolation properties between the full and reduced transfer functions can be attained at prescribed points. Note that I_m and I_p stand for $m \times m$ and $p \times p$ identity matrices, respectively.

LEMMA 3.1. *Let $\mu \in \mathbb{C}$ be a point that does not belong to the spectrum of $L(s) = A - sE$. Furthermore, let $\mathcal{W}_{\mathrm{up}} = \mathcal{W} \oplus \mathcal{W}_\mu$ and $\mathcal{V}_{\mathrm{up}} = \mathcal{V} \oplus \mathcal{V}_\mu$ for two subspaces $\mathcal{V}, \mathcal{W}$ of equal dimension, and $\mathcal{V}_\mu$, $\mathcal{W}_\mu$ defined as*

$$\mathcal{V}_\mu := \bigoplus_{j=0}^{q} \mathrm{Ran}\left[\left(\left\{(A-\mu E)^{-1}E\right\}^j (A-\mu E)^{-1}B\right) P_R(\mu)\right],$$

$$\mathcal{W}_\mu := \bigoplus_{j=0}^{q} \mathrm{Ran}\left[\left(C(A-\mu E)^{-1}\left\{E(A-\mu E)^{-1}\right\}^j\right)^* P_L(\mu)\right]$$

for some integer $q \geq 0$, and

$$P_R(\mu) = \begin{cases} I_m & \text{if } m \leq p \\ H(\mu)^* & \text{if } m > p \end{cases}, \qquad P_L(\mu) = \begin{cases} I_p & \text{if } p \leq m \\ H(\mu) & \text{if } p > m \end{cases}. \tag{3.2}$$

If μ is not a pole of $H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}$, then we have

1. $H(\mu) = H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}(\mu)$,
2. $H^{(j)}(\mu) = \left[H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}\right]^{(j)}(\mu)$ *for* $j = 1, \ldots, q$, *and*
3. $P_L(\mu)^* H^{(j)}(\mu) P_R(\mu) = P_L(\mu)^* \left[H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}\right]^{(j)}(\mu) P_R(\mu)$ *for* $j = q{+}1, \ldots, 2q{+}1$,

where $H^{(j)}$, $\left[H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}\right]^{(j)}$ denote the jth derivatives of H, $H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}$, respectively.

Proof. For every $j \in \{0, \ldots, q\}$, we have

$$\frac{d^j}{ds^j}\left\{(A-sE)^{-1}B\right\}\Big|_{s=\mu} = \left\{(A-\mu E)^{-1}E\right\}^j (A-\mu E)^{-1}B, \quad \text{and}$$

$$\frac{d^j}{ds^j}\left\{C(A-sE)^{-1}\right\}\Big|_{s=\mu} = C(A-\mu E)^{-1}\left\{E(A-\mu E)^{-1}\right\}^j.$$

Hence, letting $r := \min\{p, q\}$, for every $v, w \in \mathbb{C}^r$ it follows from the definitions of $\mathcal{V}_m$ and $\mathcal{W}_m$ that

$$\left(\frac{d^j}{ds^j}\left\{(A-sE)^{-1}B\right\}\Big|_{s=\mu}\right) P_R(\mu)v \in \mathcal{V}_\mu, \quad \text{and}$$

$$\left(\frac{d^j}{ds^j}\left\{C(A-sE)^{-1}\right\}\Big|_{s=\mu}\right)^* P_L(\mu)w \in \mathcal{W}_\mu$$

for $j = 0, 1, \ldots, q$. Now [7, Theorem 1] implies that for every $v, w \in \mathbb{C}^r$ we have

$$H^{(j)}(\mu)\,(P_R(\mu)v) \;=\; \left[H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}\right]^{(j)}(\mu)\,(P_R(\mu)v) \tag{3.3}$$

$$(P_L(\mu)w)^*\,H^{(j)}(\mu) \;=\; (P_L(\mu)w)^*\left[H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}\right]^{(j)}(\mu) \tag{3.4}$$

for $j = 0, 1, \ldots, q$, and

$$(P_L(\mu)w)^*\,H^{(j)}(\mu)\,(P_R(\mu)v) \;=\; (P_L(\mu)w)^*\left[H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}\right]^{(j)}(\mu)\,(P_R(\mu)v) \tag{3.5}$$

for $j = 0, 1, \ldots, 2q+1$.

We consider the cases $m = p$, $m > p$ and $m < p$ separately below. Throughout the rest of this proof, recall that $r := \min\{p, q\}$.

Case 1 ($m = p$) : In this case, $P_R(\mu)$ and $P_L(\mu)$ are the identity matrices. Let $j \in \{0, \ldots, 2q+1\}$. For every $v, w \in \mathbb{C}^r$, we deduce from (3.5) that

$$w^*\,H^{(j)}(\mu)\,v \;=\; w^*\left[H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}\right]^{(j)}(\mu)\,v$$

holds for every $v, w \in \mathbb{C}^r$. This in turn implies $H^{(j)}(\mu) = \left[H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}\right]^{(j)}(\mu)$.

Case 2 ($m > p$) : In this case, $P_R(\mu) = H(\mu)^*$ and $P_L(\mu) = I_p$. For $j = 0, 1, \ldots, q$, denoting with e_ℓ the ℓth column of I_p, it follows from (3.4) that

$$e_\ell^*\,H^{(j)}(\mu) \;=\; e_\ell^*\left[H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}\right]^{(j)}(\mu)$$

for $\ell = 0, 1, \ldots, p$ so that $H^{(j)}(\mu) = \left[H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}\right]^{(j)}(\mu)$.

Now let $j \in \{q+1, \ldots, 2q+1\}$. For every $v, w \in \mathbb{C}^r$, by (3.5), we have

$$w^*P_L(\mu)^*H^{(j)}(\mu)P_R(\mu)v \;=\; w^*P_L(\mu)^*\left[H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}\right]^{(j)}(\mu)P_R(\mu)v. \tag{3.6}$$

This shows that $P_L(\mu)^*H^{(j)}(\mu)P_R(\mu) = P_L(\mu)^*\left[H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}\right]^{(j)}(\mu)P_R(\mu)$ as desired.

Case 3 ($m < p$) : As $P_R(\mu) = I_m$ and $P_L(\mu) = H(\mu)$, for $j = 0, 1, \ldots, q$ the identity in (3.3) implies

$$H^{(j)}(\mu)\,e_\ell \;=\; \left[H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}\right]^{(j)}(\mu)\,e_\ell$$

for $\ell = 0, 1, \ldots, m$, where e_ℓ is the ℓth column of I_m. Consequently, we deduce $H^{(j)}(\mu) = \left[H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}\right]^{(j)}(\mu)$ for $j = 0, 1, \ldots, q$.

For $j \in \{q+1, \ldots, 2q+1\}$, once again the equality in (3.6) holds for every $v, w \in \mathbb{C}^r$ by (3.5), implying $P_L(\mu)^*H^{(j)}(\mu)P_R(\mu) = P_L(\mu)^*\left[H_{\mathsf{red}}^{\mathcal{W}_{\mathrm{up}},\mathcal{V}_{\mathrm{up}}}\right]^{(j)}(\mu)P_R(\mu)$. ∎

The result above holds even when $P_R(\mu)$ and $P_L(\mu)$ are replaced by the identity matrices of proper sizes, but then, unless $m = p$, the matrices $(\,\{(A-\mu E)^{-1}E\}^j(A-\mu E)^{-1}B\,)$ and $(\,C(A-\mu E)^{-1}\{E(A-\mu E)^{-1}\}^j\,)^*$ do not have equal number of columns. Hence, the role of $P_R(\mu)$ and $P_L(\mu)$ is to make sure $\mathcal{V}_m$ and $\mathcal{W}_m$ are defined in terms of matrices with equal number of columns and of usually full column rank so that the dimensions of $\mathcal{V}_m$ and $\mathcal{W}_m$ are usually the same.

Our proposed framework, built on these projection and interpolation ideas, operates as follows. It computes the dominant poles of a reduced system at every iteration. As the reduced systems are of low order, in practice this is achieved by first computing all of the finite poles of the reduced system, for instance by using the QZ algorithm, and sorting them from the largest to the smallest based on the dominance metric in

(1.5). Then, the framework expands the subspaces so that the transfer function of the reduced system after expansion Hermite interpolates the transfer function of the full system at the computed dominant poles of the reduced system before expansion. For this interpolatory subspace expansion task, we put Lemma 3.1 in use. The details of the proposed framework are given in Algorithm 3.1, where the dominant poles of the reduced system are computed in line 3, and the subspaces are expanded in lines 5-15 so as to satisfy Hermite interpolation at the points selected among these computed dominant poles of the reduced system based on whether they are yet to converge to actual poles. We postpone the discussions of how the initial subspaces are constructed in line 1, the termination condition to assert convergence in line 4, and the criterion to decide whether the estimates $\lambda_\ell^{(j)}, v_\ell^{(j)}$ have converged in line 7 to Section 5. A subtle issue is that we require the interpolation parameter q in Algorithm 3.1 to satisfy $q \geq 1$ if $m = p$ and $q \geq 2$ otherwise. These choices of q (by Lemma 3.1) ensure that the first two derivatives of the full transfer function are interpolated by those of the reduced transfer function at the interpolation points, which is exploited by the quadratic convergence proof in the next section for Algorithm 3.1.

The framework resembles the one that we have introduced for $\mathcal{L}_\infty$-norm computation in [1]. Only here the dominant poles of the reduced systems are used as the interpolation points, whereas in [1] the interpolation points are chosen on the imaginary axis as the points where the $\mathcal{L}_\infty$ norm of the reduced system is attained. A second remark is that rather than interpolating at all dominant poles of the reduced system, we could interpolate at only one of the dominant poles at every iteration, e.g., the most dominant one among the dominant poles of the reduced system that are yet to converge. We have explored such alternatives in [6] in the context of computing a few nonlinear eigenvalues closest to a prescribed target. Our numerical experience is that interpolating at all eigenvalues of the reduced problem is usually more reliable, even though in some cases one-per-subspace-iteration interpolation strategy may be more efficient.

Another possible variation of the subspace framework outlined in Algorithm 3.1 is setting the left-hand projection subspace equal to the right-hand subspace, which is commonly referred as a *one-sided* subspace framework in model reduction. In contrast, a subspace framework such as Algorithm 3.1 that employs different subspaces from left and right is referred as a *two-sided* subspace framework. In a one-sided variation of Algorithm 3.1, the right-hand subspace $\mathcal{V}_\ell$ at iteration ℓ is formed as in Algorithm 3.1, however $\mathcal{W}_\ell = \mathcal{V}_\ell$. An analogue of the Hermite interpolation result (i.e., Lemma 3.1) still holds; full and reduced transfer functions are equal at the interpolation point, but their derivatives match only up to the qth derivative. In the one-sided setting, provided $q \geq 2$, the rate-of-convergence analysis in the next section still applies leading to the quick convergence result in Theorem 4.3. The advantages of a one-sided variation are that an orthonormal basis for only one subspace needs to be kept, and, with $q = 2$, fewer number of back and forward substitutions per iteration may be required while retaining quick convergence. However, in our experience, it is numerically less stable than its two-sided counterpart.

4. Rate of Convergence of the Subspace Framework. We consider Algorithm 3.1 when only the dominant pole is sought (i.e., with $\kappa = 1$). Furthermore, without loss of generality, let us assume λ_1 in (1.4) is the dominant pole of system (1.1). To simplify the notation, we set $\mu_\ell := \lambda_\ell^{(1)}$ for the sequence $\{\lambda_\ell^{(1)}\}$ generated by the algorithm, and investigate how small $|\mu_{j+1} - \lambda_1|$ as compared to $|\mu_j - \lambda_1|$ assuming μ_j and μ_{j+1} are sufficiently close to λ_1.

Algorithm 3.1 Subspace framework to compute dominant poles of (1.1)

Input: system matrices $A, E \in \mathbb{C}^{n\times n}$, $B \in \mathbb{C}^{n\times m}$, $C \in \mathbb{C}^{p\times n}$, $D \in \mathbb{C}^{p\times m}$ for the descriptor system in (1.1), interpolation parameter $q \in \mathbb{Z}$ such that $q \geq 1$ if $m = p$ and $q \geq 2$ if $m \neq p$, and number of dominant poles sought κ

Output: estimate $\zeta_j \in \mathbb{C}$ for the jth dominant pole of (1.1) and the corresponding eigenvector estimate z_j of $L(s) = A - sE$ for $j = 1, 2, \cdots, \kappa$

1: % **form the initial subspaces**
Set matrices V_0, W_0 whose columns form orthonormal bases for initial subspaces.

2: **for** $\ell = 1, 2, \ldots$ **do**

3: % **update the estimates for the dominant poles**
$\lambda_\ell^{(j)}, v_\ell^{(j)} \leftarrow j$th dominant pole of the system with transfer func. $H_{\mathsf{red}}^{\mathcal{W}_{\ell-1},\mathcal{V}_{\ell-1}}(s)$ and corresponding eigenvector of $L^{W_{\ell-1},V_{\ell-1}}(s) = W_{\ell-1}^* A V_{\ell-1} - s W_{\ell-1}^* E V_{\ell-1}$ for $j = 1, \ldots, \kappa$, where $\mathcal{W}_{\ell-1}, \mathcal{V}_{\ell-1}$ are the subspaces spanned by the columns of $W_{\ell-1}$, $V_{\ell-1}$.

4: % **terminate in the case of convergence**
Return $\zeta_j \leftarrow \lambda_\ell^{(j)}$, $z_j \leftarrow V_{\ell-1} v_\ell^{(j)}$ for $j = 1, \ldots, \kappa$ if convergence occurred.

5: % **lines 5-15: expand subspaces to interpolate at** $\lambda_\ell^{(j)}$, $j = 1, \ldots, \kappa$
$V_\ell \leftarrow V_{\ell-1}$ and $W_\ell \leftarrow W_{\ell-1}$.

6: **for** $j = 1, \ldots, \kappa$ **do**

7: **if** $\lambda_\ell^{(j)}, v_\ell^{(j)}$ did not converge up to the prescribed tolerance **then**

8: $\widehat{V} \leftarrow (A - \lambda_\ell^{(j)} E)^{-1} B P_R(\lambda_\ell^{(j)})$, $\widetilde{V} \leftarrow \widehat{V}$, $\widehat{W} \leftarrow (A - \lambda_\ell^{(j)} E)^{-*} C^* P_L(\lambda_\ell^{(j)})$ and $\widetilde{W} \leftarrow \widehat{W}$, where $P_R(\lambda_\ell^{(j)}), P_L(\lambda_\ell^{(j)})$ are as in (3.2).

9: **for** $r = 1, \ldots, q$ **do**

10: $\widehat{V} \leftarrow (A - \lambda_\ell^{(j)} E)^{-1} E \widehat{V}$ and $\widetilde{V} \leftarrow \begin{bmatrix} \widetilde{V} & \widehat{V} \end{bmatrix}$.

11: $\widehat{W} \leftarrow (A - \lambda_\ell^{(j)} E)^{-*} E^* \widehat{W}$ and $\widetilde{W} \leftarrow \begin{bmatrix} \widetilde{W} & \widehat{W} \end{bmatrix}$.

12: **end for**

13: $V_\ell \leftarrow \mathrm{orth}\left(\begin{bmatrix} V_\ell & \widetilde{V} \end{bmatrix}\right)$ and $W_\ell \leftarrow \mathrm{orth}\left(\begin{bmatrix} W_\ell & \widetilde{W} \end{bmatrix}\right)$.

14: **end if**

15: **end for**

16: **end for**

Our analysis operates on the function

$$f(s) := \frac{|\det(sE - A)|^2}{\|C \cdot \mathrm{adj}(sE - A) \cdot B\|_F^2} \tag{4.1}$$

and its reduced counterpart at the end of the jth subspace iteration, that is

$$f_j(s) := \frac{|\det(sW_j^* E V_j - W_j^* A V_j)|^2}{\|CV_j \cdot \mathrm{adj}(sW_j^* E V_j - W_j^* A V_j) \cdot W_j^* B\|_F^2}, \tag{4.2}$$

where $\mathrm{adj}(\cdot)$ denotes the adjugate of its matrix argument. Clearly, using the notation $\Lambda(F, G)$ for the set of finite eigenvalues of the pencil $L(s) = F - sG$, we have

$$f(s) = \frac{1}{\|H(s)\|_F^2} \ \forall s \notin \Lambda(A, E), \quad f_j(s) = \frac{1}{\|H_{\mathsf{red}}^{\mathcal{W}_j,\mathcal{V}_j}(s)\|_F^2} \ \forall s \notin \Lambda(W_j^* A V_j, W_j^* E V_j).$$

Moreover, $f(s)$ and $f_j(s)$ are well-defined under mild assumptions even at the finite eigenvalues of the associated pencils. For instance, as we show next, unless the left eigenspace associated with λ_1 is orthogonal to $\mathrm{Ran}(B)$ (i.e., unless λ_1 is uncontrollable), or the right eigenspace associated with λ_1 is orthogonal to $\mathrm{Ran}(C^*)$ (i.e., unless λ_1 is unobservable), $f(s)$ is well-defined at λ_1. To see this, let us consider

$$\mathrm{adj}(sE-A) \;=\; \det(sE-A)(sE-A)^{-1} \tag{4.3}$$

near λ_1. Observe that $\mathrm{adj}(sE-A)$ is continuous everywhere, in particular at $s=\lambda_1$. Hence, by taking the limits of both sides in (4.3) as $s\to\lambda_1$ and recalling the Kronecker canonical form (1.3), it follows that

$$C\,\mathrm{adj}(\lambda_1 E-A)\,B \;=\; (-1)^{n-\widetilde{n}}\cdot\det(W^{-*}V^{-1})\cdot CV\begin{bmatrix}\prod_{j=2}^{\widetilde{n}}\lambda_1-\lambda_j & 0\\ 0 & 0\end{bmatrix}W^*B.$$

From here we deduce that, unless $w_1\perp\mathrm{Ran}(B)$ or $v_1\perp\mathrm{Ran}(C^*)$, the denominator in (4.1) is nonzero at $s=\lambda_1$. Similarly, if μ_{j+1} is a controllable and observable pole of the reduced transfer function $H_{\mathsf{red}}^{\mathcal{W}_j,\mathcal{V}_j}$, then $f_j(s)$ is well-defined at μ_{j+1}. To summarize, simple conditions, such as the minimality of the original system and the reduced system at the end of the jth subspace iteration (as the minimality implies both the controllability and the observability of all poles), guarantee the well-posedness of $f(s)$ and $f_j(s)$ everywhere. We make the following assumptions that guarantee the well-posedness of $f(s)$ and $f_j(s)$ at λ_1 and μ_{j+1} throughout the rest of this section.

Assumption 4.1. *The following conditions are satisfied:*

1. $C\cdot\mathrm{adj}(\lambda_1E-A)\cdot B\neq 0$.
2. *For prescribed $\beta>0$, we have* $\|CV_j\cdot\mathrm{adj}(\mu_{j+1}W_j^*EV_j-W_j^*AV_j)\cdot W_j^*B\|_2\geq\beta$.

It is evident that the numerators and denominators in (4.1) and (4.2) defining f and f_j are polynomials in $\Re s$ and $\Im s$, the real and imaginary parts of s. It can be shown by also employing Assumption 4.1 that these functions are three-times differentiable with respect to $\Re s$, $\Im s$ with all of their first three derivatives bounded in a ball $B(\lambda_1,\delta):=\{z\in\mathbb{C}\,|\,|z-\lambda_1|\leq\delta\}$ that contain μ_j, μ_{j+1} (recall that μ_j, μ_{j+1} are assumed to be sufficiently close to λ_1). Indeed, it can be shown that the radius δ and bounds on the derivatives of f_j are uniform over all V_j, W_j as long as the second condition in Assumption 4.1 is met. The arguments similar to those in [1, Section 3] can be used to show the existence of the ball, and the uniformity of the bounds.

The interpolatory properties between $f(s)$ and $f_j(s)$ and their first two derivatives at μ_j follow from Lemma 3.1. In the subsequent arguments, we use the notations

$$f'(\widehat{s}) \;=\; \begin{bmatrix}\frac{\partial f}{\partial\Re s}(\widehat{s}) & \frac{\partial f}{\partial\Im s}(\widehat{s})\end{bmatrix}^T,\quad f_j'(\widehat{s}) \;=\; \begin{bmatrix}\frac{\partial f_j}{\partial\Re s}(\widehat{s}) & \frac{\partial f_j}{\partial\Im s}(\widehat{s})\end{bmatrix}^T$$

$$\nabla^2 f(\widehat{s}) \;=\; \begin{bmatrix}\frac{\partial^2 f}{\partial\Re s\,\partial\Re s}(\widehat{s}) & \frac{\partial^2 f}{\partial\Re s\,\partial\Im s}(\widehat{s})\\ \frac{\partial^2 f}{\partial\Im s\,\partial\Re s}(\widehat{s}) & \frac{\partial^2 f}{\partial\Im s\,\partial\Im s}(\widehat{s})\end{bmatrix},\quad \nabla^2 f_j(\widehat{s}) \;=\; \begin{bmatrix}\frac{\partial^2 f_j}{\partial\Re s\,\partial\Re s}(\widehat{s}) & \frac{\partial^2 f_j}{\partial\Re s\,\partial\Im s}(\widehat{s})\\ \frac{\partial^2 f_j}{\partial\Im s\,\partial\Re s}(\widehat{s}) & \frac{\partial^2 f_j}{\partial\Im s\,\partial\Im s}(\widehat{s})\end{bmatrix}$$

at a given $\widehat{s}\in\mathbb{C}$, where f and f_j are twice differentiable with respect to $\Re s$, $\Im s$. Moreover, $\mathcal{R}:\mathbb{C}\to\mathbb{R}^2$ is the linear map $\mathcal{R}(z):=\begin{bmatrix}\Re z & \Im z\end{bmatrix}^T$.

Theorem 4.2. *Suppose μ_j is not an eigenvalue of $L(s)=A-sE$ and not a pole of $H_{\mathsf{red}}^{\mathcal{W}_j,\mathcal{V}_j}$, and Assumption 4.1 holds. Then*

$$f'(\mu_j) \;=\; f_j'(\mu_j) \qquad \textit{and} \qquad \nabla^2 f(\mu_j) \;=\; \nabla^2 f_j(\mu_j). \tag{4.4}$$

Proof. We deduce from Lemma 3.1 that

$$H(\mu_j) = H_{\mathsf{red}}^{\mathcal{W}_j,\mathcal{V}_j}(\mu_j), \quad H'(\mu_j) = [H_{\mathsf{red}}^{\mathcal{W}_j,\mathcal{V}_j}]'(\mu_j), \quad H''(\mu_j) = [H_{\mathsf{red}}^{\mathcal{W}_j,\mathcal{V}_j}]''(\mu_j). \tag{4.5}$$

The result is obtained by differentiating

$$f(s) \;=\; \frac{1}{\mathrm{Trace}(H(s)^*H(s))}, \quad f_j(s) \;=\; \frac{1}{\mathrm{Trace}(H_{\mathsf{red}}^{\mathcal{W}_j,\mathcal{V}_j}(s)^*H_{\mathsf{red}}^{\mathcal{W}_j,\mathcal{V}_j}(s))}$$

and employing the equalities in (4.5). ∎

We have $f(s) \geq 0$ and $f_j(s) \geq 0$ for all $s \in B(\lambda_1, \delta)$ by the defining equations in (4.1) and (4.2), as well as $f(\lambda_1) = f_j(\mu_{j+1}) = 0$. Consequently, λ_1, μ_{j+1} are global minimizers of f, f_j implying $f'(\lambda_1) = f_j'(\mu_{j+1}) = 0$. It follows that

$$\begin{aligned}
0 \;=\; f'(\lambda_1) \;&=\; f'(\mu_j) + \int_0^1 \nabla^2 f(\mu_j + t(\lambda_1 - \mu_j))\mathcal{R}(\lambda_1 - \mu_j)\,\mathrm{d}t \\
&=\; f'(\mu_j) + \nabla^2 f(\mu_j)\mathcal{R}(\lambda_1 - \mu_j) \\
&\qquad + \int_0^1 \left\{\nabla^2 f(\mu_j + t(\lambda_1 - \mu_j)) - \nabla^2 f(\mu_j)\right\}\mathcal{R}(\lambda_1 - \mu_j)\,\mathrm{d}t \\
&=\; f_j'(\mu_j) + \nabla^2 f_j(\mu_j)\mathcal{R}(\lambda_1 - \mu_j) \\
&\qquad + \int_0^1 \left\{\nabla^2 f(\mu_j + t(\lambda_1 - \mu_j)) - \nabla^2 f(\mu_j)\right\}\mathcal{R}(\lambda_1 - \mu_j)\,\mathrm{d}t,
\end{aligned}$$

where the last equality is due to (4.4). An application of Taylor's theorem with second order remainder to $f_j'(s)$ at μ_{j+1} about μ_j together with $f_j'(\mu_{j+1}) = 0$ yield

$$\begin{aligned}
f_j'(\mu_j) + \nabla^2 f_j(\mu_j)\mathcal{R}(\lambda_1 - \mu_j) \;&=\; \nabla^2 f_j(\mu_j)\mathcal{R}(\lambda_1 - \mu_{j+1}) + \mathcal{O}(|\mu_{j+1} - \mu_j|^2) \\
&=\; \nabla^2 f(\mu_j)\mathcal{R}(\lambda_1 - \mu_{j+1}) + \mathcal{O}(|\mu_{j+1} - \mu_j|^2).
\end{aligned}$$

Substituting the right-hand side of the last equality above in the previous equation, we deduce

$$\begin{aligned}
\nabla^2 f(\mu_j)\mathcal{R}(\mu_{j+1} - \lambda_1) = \int_0^1 \left\{\nabla^2 f(\mu_j + t(\lambda_1 - \mu_j)) - \nabla^2 f(\mu_j)\right\}\mathcal{R}(\lambda_1 - \mu_j)\mathrm{d}t \\
+\; \mathcal{O}(|\mu_{j+1} - \mu_j|^2).
\end{aligned} \tag{4.6}$$

Let us also suppose $\nabla^2 f(\lambda_1)$ is invertible. Then, by the continuity of $\nabla^2 f$ in $B(\lambda_1, \delta)$ and letting $\sigma_{\min}(\nabla^2 f(s))$ denote the smallest singular value of $\nabla^2 f(s)$, there exists a constant $\eta > 0$ such that

$$\sigma_{\min}(\nabla^2 f(s)) \geq \eta \qquad \forall s \in B(\lambda_1, \delta), \tag{4.7}$$

by choosing δ even smaller if necessary. Furthermore, boundedness of the third derivatives of f implies the Lipschitz continuity of its second derivatives, in particular the existence of a constant $\gamma > 0$ such that

$$\|\nabla^2 f(\mu_j + t(\lambda_1 - \mu_j)) - \nabla^2 f(\mu_j)\|_2 \leq \gamma t|\lambda_1 - \mu_j| \qquad \forall t \in [0, 1]. \tag{4.8}$$

By taking the 2-norms of both sides in (4.6), using the triangle inequality, as well as the inequalities (4.7), (4.8), we obtain

$$\eta\,|\lambda_1-\mu_{j+1}| \;\leq\; (\gamma/2)\,|\lambda_1-\mu_j|^2 \;+\; \mathcal{O}(|\mu_{j+1}-\mu_j|^2).$$

Finally, noting that $\mathcal{O}(|\mu_{j+1}-\mu_j|^2)$ terms are bounded from above by $c\,|\mu_{j+1}-\mu_j|^2 \leq 2c\{|\lambda_1-\mu_j|^2+|\lambda_1-\mu_{j+1}|^2\} \leq 2c\{|\lambda_1-\mu_j|^2+\delta\,|\lambda_1-\mu_{j+1}|\}$ for some constant c, we have

$$(\eta-2c\delta)\,|\lambda_1-\mu_{j+1}| \;\leq\; (\gamma/2+2c)\,|\lambda_1-\mu_j|^2\,.$$

Our findings are summarized in the next result.

THEOREM 4.3 (At Least Quadratic Convergence of the Subspace Framework). *Assuming that the iterates μ_j, μ_{j+1} of Algorithm 3.1 are sufficiently close to λ_1, the Hessian $\nabla^2 f(\lambda_1)$ is invertible, and the assumptions of Theorem 4.2 hold, we have*

$$|\lambda_1-\mu_{j+1}| \;\leq\; C\,|\lambda_1-\mu_j|^2$$

for some constant $C>0$.

5. Practical Details. Here, we spell out some details that one has to take into consideration in a practical implementation of Algorithm 3.1.

5.1. Forming Initial Subspaces. Initial subspaces in line 1 of Algorithm 3.1 are constructed so as to attain Hermite interpolation between the full and reduced system at prescribed points $\rho_1,\ldots,\rho_\varphi \in \mathbb{C}$. It is desirable that $\rho_1,\ldots,\rho_\varphi$ are not very far away from the dominant poles of H.

One possibility for the selection of these initial interpolation points is to form a rational approximation for the full transfer function H, then use the dominant poles of the rational approximation. The AAA algorithm [26] is widely employed at the moment for such rational approximation problems. We have attempted to approximate the entries of H using the AAA algorithm, but, on benchmark examples, this approach usually did not yield points close to the dominant poles.

Instead, we apply the algorithm in [1] for $\mathcal{L}_\infty$-norm computation crudely, which gives rise to a reduced order model that approximates H reasonably well on some parts of the imaginary axis where it exhibits large peaks. The initial interpolation points $\rho_1,\ldots,\rho_\varphi$ are then set equal to the dominant poles of the retrieved reduced order model.

5.2. Termination Condition. A natural choice for a termination condition in line 4 of Algorithm 3.1 is based on the ∞-norms of the residuals

$$\mathsf{Rs}(\lambda_\ell^{(j)}, v_\ell^{(j)}) \;:=\; \|(A-\lambda_\ell^{(j)}E)\cdot V_{\ell-1}v_\ell^{(j)}\|_\infty \tag{5.1}$$

for $j=1,\ldots,\kappa$. If all of $\mathsf{Rs}(\lambda_\ell^{(j)}, v_\ell^{(j)})$ are smaller than a prescribed tolerance `tol`, then we terminate with $\zeta_j=\lambda_\ell^{(j)}$, $z_j=V_{\ell-1}v_\ell^{(j)}$ for $j=1,\ldots,\kappa$.

5.3. Deciding the Convergence of a Dominant Pole Estimate. The decision whether the estimates $\lambda_\ell^{(j)}, v_\ell^{(j)}$ have converged or not in line 7 of Algorithm 3.1 is also given based on the residual $\mathsf{Rs}(\lambda_\ell^{(j)}, v_\ell^{(j)})$ in (5.1). Specifically, if $\mathsf{Rs}(\lambda_\ell^{(j)}, v_\ell^{(j)}) <$ `tol`, then $\lambda_\ell^{(j)}, v_\ell^{(j)}$ are deemed as already converged, and, as a result, the subspace expansion to Hermite interpolate at $\lambda_\ell^{(j)}$ is skipped. The tolerance `tol` used here is the same as the one used for termination above.

5.4. Solutions of Linear Systems. The main computational burden of Algorithm 3.1 is due to the solutions of linear systems. Computation of the dominant poles for the reduced problems in line 3 by the QZ algorithm and orthonormalization of the bases for the projection subspaces in line 13 are quite negligible compared to the solutions of linear systems.

At each subspace iteration, several linear systems need to be solved in lines 6 - 15 of Algorithm 3.1. In this part of the algorithm and inside the for loop over j, the coefficient matrix for all of the linear systems is either $(A - \lambda_\ell^{(j)} E)$ or $(A - \lambda_\ell^{(j)} E)^*$. We compute an LU factorization of $(A - \lambda_\ell^{(j)} E)$ only once for solving all of these linear systems. Then the resulting triangular linear systems are solved.

5.5. Orthonormalization of the Bases for the Subspaces. As $\lambda_\ell^{(j)}$ tends to converge with respect to ℓ, the new directions $\widetilde{V}, \widetilde{W}$ to be included in the subspaces $\mathcal{V}_{\ell-1}, \mathcal{W}_{\ell-1}$ nearly lie in these subspaces. Hence, without orthonormalization, the matrices $[V_{\ell-1}\ \widetilde{V}]$ and $[W_{\ell-1}\ \widetilde{W}]$ would be ill-conditioned, which in turn may result in reduced order systems that are not computed accurately due to the rounding errors. Orthonormalization resolves this numerical difficulty; in particular, the matrices V_ℓ, W_ℓ with orthonormal columns at the end of the ℓth iteration are well-conditioned.

In practice, in line 13 of Algorithm 3.1, when orthogonalizing the columns of $\widetilde{V}, \widetilde{W}$ with respect to the spaces spanned by the columns of V_ℓ, W_ℓ, we first apply

$$\widetilde{V} \longleftarrow \widetilde{V} - V_\ell (V_\ell^* \widetilde{V}) \quad \text{and} \quad \widetilde{W} \longleftarrow \widetilde{W} - W_\ell (W_\ell^* \widetilde{W}) \tag{5.2}$$

several times. Then the columns of $\widetilde{V}, \widetilde{W}$ are orthonormalized, and V_ℓ, W_ℓ are augmented with these orthonormalized matrices. The reorthogonalization strategy (i.e., application of (5.2) several times) improves the accuracy in the presence of rounding errors of the columns of V_ℓ, W_ℓ as orthonormal bases for the subspaces $\mathcal{V}_\ell, \mathcal{W}_\ell$.

5.6. Complexity. Putting the formation of initial subspaces aside, there are three main computational tasks that an actual implementation of Algorithm 3.1 must carry out at every subspace iteration (i.e., per an iteration of the outer for loop).

1. Computation of dominant poles of the reduced system in line 3.
2. Solutions of linear systems in lines 8, 10, 11.
3. Orthonormalizing the bases for projection subspaces in line 13.

As argued above the computational costs for items 1. and 3. are quite negligible. If the subspace dimensions are d, which is typically much smaller than n, then 1. requires an application of the QZ algorithm at a cost of $\mathcal{O}(d^3)$. On the other hand, the total cost due to 3. is $\mathcal{O}(nd^2)$ for each one of the at most κ dominant poles, as line 13 involves two orthonormalizations with respect to $n \times d$ matrices with orthonormal columns. The constant hidden in this last $\mathcal{O}(\cdot)$ notation is small.

As for item 2., at the ℓth subspace iteration, for each dominant pole estimate μ, an LU factorization of the $n \times n$ matrix $A - \mu E$ is computed once, and $2\min\{m, p\}(q+1)$ back and forward substitutions are applied to $n \times n$ triangular systems. Hence, taking all dominant pole estimates into account, at most κ LU factorizations, $2\kappa \min\{m, p\}(q+1)$ forward and back substitutions on systems of size $n \times n$ need to be performed per subspace iteration. These tasks related to 2. typically dominate the computations, and determine the runtime. If the system at hand is such that LU factorizations, back and forward substitutions can be carried out in linear time, say at a cost of cn for a small constant c, then orthonormalization costs may also be influential, but most often this is not the case. Usually, LU factorizations are the most dominant factor.

The number of forward and back substitutions increases linearly with respect to m and p, so, for a system with many inputs and outputs, time required for forward and back substitutions may be significant, perhaps as significant as LU factorizations. The number of forward and back substitutions increases also linearly with respect to q, the parameter that determines the number of derivatives to be interpolated, yet in practice we choose q small (e.g., $q = 1$), as this already ensures quadratic convergence.

6. Numerical Results. We have implemented Algorithm 3.1 in Matlab taking the practical issues in Section 5 into account. Here, we perform numerical experiments with it in Matlab 2020b on an an iMac with Mac OS 12.1 operating system, Intel® Core™ i5-9600K CPU and 32GB RAM.

Throughout, our implementation terminates when $\mathsf{Rs}(\lambda_\ell^{(j)}, v_\ell^{(j)}) < \texttt{tol}$ for $j = 1, \ldots, \kappa$ for the tolerance $\texttt{tol} = 10^{-7}$ unless otherwise specified, where κ is the number of dominant poles prescribed, and the residual $\mathsf{Rs}(\lambda_\ell^{(j)}, v_\ell^{(j)})$ is as in (5.1). As discussed in Section 5.1, the initial subspaces $\mathcal{V}_0$, $\mathcal{W}_0$ are chosen so that Hermite interpolation is attained at the ten most dominant poles of a crude reduced model obtained by applying the algorithm in [1], excluding Section 6.3 where we investigate the effect of the initial subspace dimension on the runtime. The interpolation parameter in Algorithm 3.1 and Theorem 3.1 that determines how many derivatives of the reduced transfer function and full transfer function match is set equal to $q = 1$ in all of the experiments. We follow this practice even for rectangular transfer functions with $m \neq p$. The derivation of the rapid convergence result in Section 4 applies to rectangular transfer functions when $q \geq 2$. Yet, we observe a quick convergence in the rectangular setting on benchmark examples even with $q = 1$.

We assume throughout that the descriptor system at hand as in (1.1) has real coefficient matrices A, E, B, C, D. Under this assumption, the dominant poles come in conjugate pairs, i.e., if z is one of the poles, unless z is real, its conjugate $\overline{z}$ is also a pole with exactly the same dominance metric as for z. Our implementation computes only the dominant poles with nonpositive imaginary parts; in particular, we extract the dominant poles of the reduced problem with nonpositive imaginary parts and interpolate only at these poles at every subspace iteration. In the subsequent subsections of this section, what we refer as the most dominant κ poles are indeed the most dominant κ poles among the poles with nonpositive imaginary parts. As a result, we count a conjugate pair of dominant poles only once and not twice. For instance, when we report the five most dominant poles and if those poles are not real, in reality the ten most dominant poles are computed.

In the next four subsections, we report the results by our implementation of Algorithm 3.1 on benchmark examples for dominant pole estimation made available by Joost Rommes most of which can be accessed from his website[2], as well as benchmark examples from SLICOT collection for model reduction[3]. Comparisons are provided with the subspace accelerated multiple-input-multiple-output (MIMO) dominant pole algorithm (SAMDP) [28], specifically with the implementation on Rommes' website.

6.1. Convergence and Illustration of Algorithm 3.1 on `M80PI_n`. We first illustrate the convergence of Algorithm 3.1 on the `M80PI_n` example. This is an example with $n = 4182$ and $m = p = 3$. When we attempt to compute the most dominant pole, it takes only two iterations to fulfill the termination condition, that is the residual of the most dominant pole estimate and the corresponding eigenvector

[2] http://sites.google.com/site/rommes/software
[3] http://slicot.org/20-site/126-benchmark-examples-for-model-reduction

TABLE 6.1
The iterates and corresponding residuals of Algorithm 3.1 to find the most dominant pole of the `M80PI_n` *example.*

ℓ	$\lambda_\ell^{(1)}$	$\mathsf{Rs}(\lambda_\ell^{(1)}, v_\ell^{(1)})$
1	$-2.885476680562\text{e+}01 - 5.073419687488\text{e+}03\text{i}$	$8.398 \cdot 10^{-4}$
2	$-2.885470736631\text{e+}01 - 5.073419627243\text{e+}03\text{i}$	$1.044 \cdot 10^{-13}$

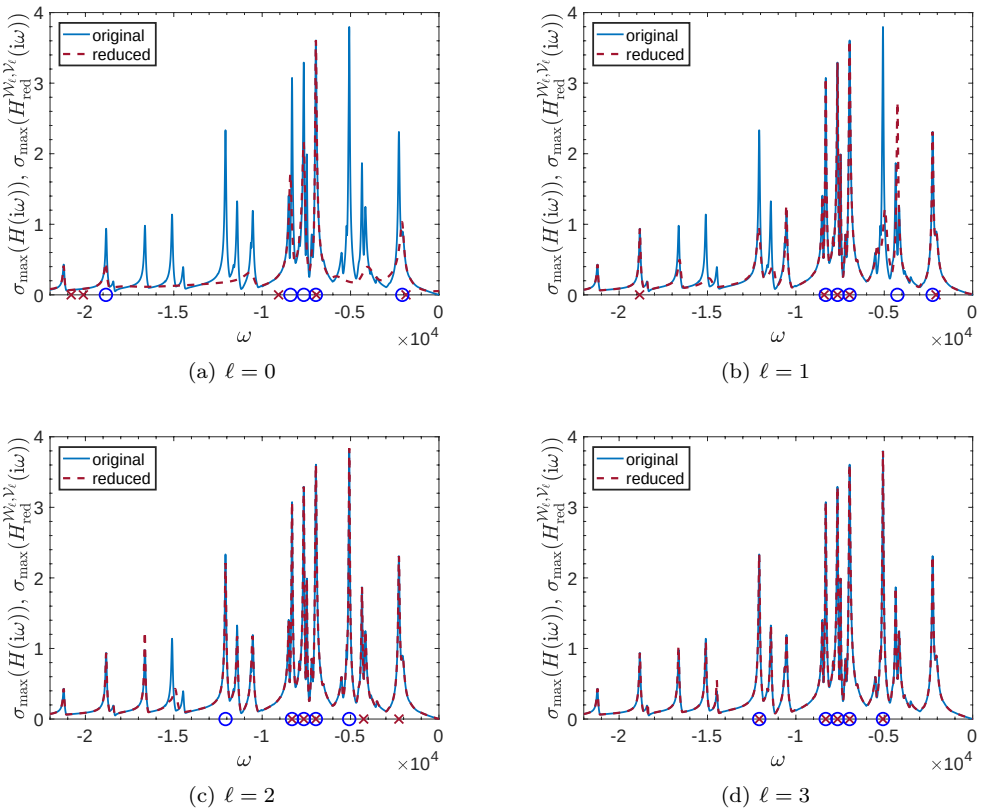
TABLE 6.2
The residuals of the iterates of Algorithm 3.1 to compute the most dominant five poles of the `M80PI_n` *example. Interpolation is performed at the iterates whose residuals are typed in blue italic.*

ℓ	$\mathsf{Rs}(\lambda_\ell^{(1)}, v_\ell^{(1)})$	$\mathsf{Rs}(\lambda_\ell^{(2)}, v_\ell^{(2)})$	$\mathsf{Rs}(\lambda_\ell^{(3)}, v_\ell^{(3)})$	$\mathsf{Rs}(\lambda_\ell^{(4)}, v_\ell^{(4)})$	$\mathsf{Rs}(\lambda_\ell^{(5)}, v_\ell^{(5)})$
1	$\mathit{8.398 \cdot 10^{-4}}$	$\mathit{3.051 \cdot 10^{-2}}$	$\mathit{2.500 \cdot 10^{-1}}$	$\mathit{8.419 \cdot 10^{-1}}$	$\mathit{1.651 \cdot 10^{-1}}$
2	$1.488 \cdot 10^{-13}$	$7.642 \cdot 10^{-14}$	$4.413 \cdot 10^{-9}$	$1.185 \cdot 10^{-8}$	$\mathit{9.539 \cdot 10^{-5}}$
3	$5.219 \cdot 10^{-14}$	$3.553 \cdot 10^{-14}$	$2.324 \cdot 10^{-9}$	$6.616 \cdot 10^{-9}$	$5.182 \cdot 10^{-13}$

estimate is less than 10^{-7} after two subspace iterations. The iterates generated and the corresponding residuals are given in Table 6.1. The progress in the iterates in the table is consistent with at-least-quadratic-convergence assertion of Theorem 4.3.

The convergence behavior is similar when we attempt to locate multiple dominant poles. In Table 6.2, the residuals for the iterates of Algorithm 3.1 are listed to estimate the most dominant five poles of the `M80PI_n` example. Only three subspace iterations suffice to satisfy the termination condition, that is all residuals are below the tolerance 10^{-7} after three iterations. In the first iteration, Hermite interpolation is performed at all of the five dominant pole estimates as the residuals are larger than 10^{-7}. At the second iteration, the most dominant four pole estimates are deemed as already converged, since their residuals are below the convergence threshold, while Hermite interpolation is imposed at the estimate for the fifth dominant pole. Notably at a dominant pole estimate where Hermite interpolation is imposed in Table 6.2, the corresponding residual decreases considerably. In general, we have observed a similar convergence behavior on other examples. Only that it sometimes happens that not much progress is achieved towards convergence in the first few iterations. But once a better global approximation of the transfer function is obtained, in particular when the dominant poles estimates are close to the actual ones, convergence is very fast.

The convergence of Algorithm 3.1 on the `M80PI_n` example is also illustrated in Figure 6.1. To illustrate the convergence better, here we start with a very crude reduced system initially; see the red dashed curve in Figure 6.1(a). (A more accurate initial reduced model is used in Table 6.1 and 6.2, as we normally apply the algorithm in [1] to construct the initial reduced system a little more rigorously, a common practice in our implementation excluding this visualization.) The initial reduced transfer function $H_{\mathsf{red}}^{\mathcal{W}_0,\mathcal{V}_0}$ in Figure 6.1(a) interpolates the original transfer function H at complex points, whose imaginary parts are marked with crosses on the horizontal axis. In the same figure, the blue circles mark the imaginary parts of the five most dominant poles of $H_{\mathsf{red}}^{\mathcal{W}_0,\mathcal{V}_0}$. These five most dominant poles are used for interpolation next, giving rise to $H_{\mathsf{red}}^{\mathcal{W}_1,\mathcal{V}_1}$, the reduced transfer function in Figure 6.1(b). Similarly, the dominant poles of $H_{\mathsf{red}}^{\mathcal{W}_1,\mathcal{V}_1}$ whose imaginary parts are marked with blue circles in Figure 6.1(b) are the next set of interpolation points, leading to the reduced transfer function $H_{\mathsf{red}}^{\mathcal{W}_2,\mathcal{V}_2}$ in Figure 6.1(c). The largest singular value of the transfer function $H_{\mathsf{red}}^{\mathcal{W}_3,\mathcal{V}_3}$ after three iterations in Figure 6.1(d) seems to capture the largest singular



(a) $\ell = 0$ (b) $\ell = 1$

(c) $\ell = 2$ (d) $\ell = 3$

FIG. 6.1. *The progress of Algorithm 3.1 on the* `M80PI_n` *example. The solid blue and red dashed curves are the plots of the largest singular values of* $H(\mathrm{i}\omega)$ *and* $H_{\mathrm{red}}^{\mathcal{W}_\ell,\mathcal{V}_\ell}(\mathrm{i}\omega)$ *as functions of* $\omega \in \mathbb{R}$. *The red crosses and blue circles on the horizontal axis mark the imaginary parts of the interpolation points employed and dominant poles of* $H_{\mathrm{red}}^{\mathcal{W}_\ell,\mathcal{V}_\ell}$, *respectively.*

value of the full transfer function over the imaginary axis already quite well, and (at least) the imaginary parts of the dominant poles of $H_{\mathrm{red}}^{\mathcal{W}_3,\mathcal{V}_3}$ and H appear to be close from the figure.

6.2. Results on Benchmark Examples. A comparison of our implementation of Algorithm 3.1 and the subspace accelerated MIMO dominant pole algorithm (SAMDP) [28] on 11 benchmark examples is provided in Table 6.3.

Parameter Values. Both algorithms are initialized with exactly the same ten points in the complex plane, namely the ten most dominant poles of the reduced system produced by the algorithm in [1]. As a result, the initial subspace dimension for Algorithm 3.1 is $20 \cdot \min\{m, p\}$, i.e., recall that $q = 1$ throughout, which means that each interpolation point requires the inclusion of a subspace of dimension $2 \cdot \min\{m, p\}$. We use the implementation of SAMDP with its default parameter settings with one exception, namely the parameter that determines how LU factorizations should be computed. To be fair, both Algorithm 3.1 and SAMDP employ the command `lu(A)`, when an LU factorization of A needs to be computed. Moreover, both algorithms solve the linear systems by exploiting the LU factorizations using the same sequence

of commands based on backslashes. The default minimum and maximum subspace dimensions for SAMDP are 1 and 10, which we rely on in the numerical experiments. This means that SAMDP restarts the projection subspaces in case they become of dimension greater than 10, and the restarted subspaces are of dimension 1. In contrast, our implementation of Algorithm 3.1 does not employ a restart strategy. Our observation is that the performance of SAMDP remains nearly the same even without the restart strategy.

Comments on the Results. For all of these examples, we estimate the five most dominant poles of the full system using these two algorithms. For the first two smaller problems (`CDplayer`, `iss`), the algorithms return exactly the same five dominant poles up to prescribed tolerances. But for all other 9 examples, Algorithm 3.1 returns consistently more dominant poles compared to SAMDP; see the numbers inside the parentheses in the columns under "Five Most Dominant Poles" in Table 6.3, which represent the dominance metrics of the computed poles as in equation (1.5). There are even examples for which three or four of the five most dominant poles by Algorithm 3.1 are more dominant than the ones computed by SAMDP; see, e.g., the `M10PI_n` example for which only the most dominant poles are the same and the remaining four differ in favor of Algorithm 3.1 when their dominance metric is taken into consideration. On smaller examples, specifically on the `CDplayer`, `iss`, `S40PI_n`, `M010PI_n` examples, we have also computed all of the poles using the QR or QZ algorithm (i.e., using `eig` in Matlab), and verified that the five most dominant poles by Algorithm 3.1 listed in Table 6.3 are indeed the five most dominant poles up to the prescribed tolerances.

The runtimes of the two algorithms listed in the last column of Table 6.3 are similar; one or the other is a little faster in some cases, but there does not appear any substantial difference in the runtimes. A better insight into the runtimes can be obtained from Table 6.4. It is apparent from this table that Algorithm 3.1 performs fewer number of LU factorizations consistently. On the other hand, SAMDP usually requires fewer number of linear system solves. One main factor that determines which algorithm has a better runtime is the computational cost of an LU factorization as compared to that of solving triangular systems. On examples where LU factorization computations are considerably expensive, it is reasonable to expect that Algorithm 3.1 would have a better runtime. But, as the benchmark examples are sparse and even nearly banded at times, it is sometimes the case that LU factorizations are cheap to obtain, only at a cost of a small constant times that of triangular systems. Such systems seem to favor SAMDP. This seems to be that case for instance with the `bips07_3078` example. The second example for which SAMDP has notably smaller runtime is the `mimo8x8_system` example with $m = p = 8$; as m and p are relatively larger, Algorithm 3.1 needs to solve quite a few triangular systems causing the difference in the runtimes. In general, the larger m and p are, the more triangular systems need to be solved and more computational work is required by Algorithm 3.1, even though it should still converge quickly in a few iterations.

In any case, the main conclusion that can be drawn from these experiments is that Algorithm 3.1 is more robust than SAMDP in converging to the most dominant poles. SAMDP, together with some of the most dominant poles, seems to converge also some of the less dominant poles.

Algorithm 3.1 vs. SAMDP with the same initial subspaces. When comparing Algorithm 3.1 and SAMDP on benchmark examples above, Algorithm 3.1 is initialized with a subspace of dimension $20 \cdot \min\{m, p\}$ to attain Hermite interpolation at ten

TABLE 6.3

A comparison of Algorithm 3.1 with SAMDP [28] on 11 benchmark examples. The numbers inside the parentheses in the columns under "Five Most Dominant Poles" are the dominance metrics of the computed poles as in equation (1.5). Moreover, for each example, number of subspace iterations performed by Algorithm 3.1 until termination is given in the column of n_{iter}.

#	example	n,m,p	n_{iter}	Five Most Dominant Poles		time in s	
				SAMDP [28]	Alg. 3.1	[28]	Alg. 3.1
1	CDplayer	120,2,2	1	1.0e+02 ·	1.0e+02 ·	0.04	0.01
				-0.0023 ± 0.2257i (2.32e+06)	-0.0023 ± 0.2257i (2.32e+06)		
				-0.1227 ± 3.0654i (3.36e+03)	-0.1227 ± 3.0654i (3.36e+03)		
				-0.0781 ± 0.7775i (5.56e+02)	-0.0781 ± 0.7775i (5.56e+02)		
				-0.1976 ± 1.9658i (2.91e+02)	-0.1976 ± 1.9658i (2.91e+02)		
				-0.0742 ± 0.7382i (2.27e+02)	-0.0742 ± 0.7382i (2.27e+02)		
2	iss	270,3,3	2	-0.0039 ± 0.7751i (1.16e−01)	-0.0039 ± 0.7751i (1.16e−01)	0.02	0.02
				-0.0100 ± 1.9920i (3.38e−02)	-0.0100 ± 1.9920i (3.38e−02)		
				-0.0424 ± 8.4808i (1.20e−02)	-0.0424 ± 8.4808i (1.20e−02)		
				-0.1899 ± 37.9851i (1.07e−02)	-0.1899 ± 37.9851i (1.07e−02)		
				-0.0462 ± 9.2336i (6.24e−03)	-0.0462 ± 9.2336i (6.24e−03)		
3	S40PI_n	2182,1,1	3	1.0e+04 ·	1.0e+04 ·	0.12	0.07
				-0.0041 ± 0.6962i (3.29e+00)	-0.0041 ± 0.6962i (3.29e+00)		
				-0.0032 ± 0.7637i (3.21e+00)	-0.0032 ± 0.7637i (3.21e+00)		
				-0.0021 ± 0.7466i (1.66e+00)	-0.0021 ± 0.7466i (1.66e+00)		
				-0.0036 ± 1.2053i (1.26e+00)	-0.0032 ± 0.4351i (1.63e+00)		
				-0.0042 ± 1.0519i (9.13e−01)	-0.0036 ± 1.2053i (1.26e+00)		
4	S80PI_n	4182,1,1	5	1.0e+04 ·	1.0e+04 ·	0.22	0.21
				-0.0041 ± 0.6965i (3.31e+00)	-0.0041 ± 0.6965i (3.31e+00)		
				-0.0032 ± 0.7642i (3.17e+00)	-0.0032 ± 0.7642i (3.17e+00)		
				-0.0021 ± 0.7474i (1.69e+00)	-0.0021 ± 0.7474i (1.69e+00)		
				-0.0036 ± 1.2073i (1.25e+00)	-0.0032 ± 0.4351i (1.63e+00)		
				-0.0043 ± 1.0532i (9.54e−01)	-0.0036 ± 1.2073i (1.25e+00)		
5	M10PI_n	682,3,3	5	1.0e+04 ·	1.0e+04 ·	0.07	0.08
				-0.0028 ± 0.5035i (3.99e+00)	-0.0028 ± 0.5035i (3.99e+00)		
				-0.0040 ± 0.2262i (2.17e+00)	-0.0032 ± 0.7530i (3.98e+00)		
				-0.0032 ± 0.4340i (1.86e+00)	-0.0028 ± 0.8126i (2.92e+00)		
				-0.0043 ± 1.0316i (1.12e+00)	-0.0041 ± 0.6905i (2.90e+00)		
				-0.0049 ± 0.4153i (9.59e−01)	-0.0036 ± 1.1670i (2.24e+00)		
6	bips07_1998	15066,4,4	6	-0.8144 ± 5.3836i (1.49e+02)	-0.8144 ± 5.3836i (1.49e+02)	8.17	5.35
				-0.3471 ± 6.6647i (3.25e+01)	-7.6146 ± 1.8625i (5.75e+01)		
				-1.0365 ± 6.3718i (3.09e+01)	-8.9412 ± 0.2616i (5.25e+01)		
				-0.7473 ± 6.2077i (1.84e+01)	-0.3471 ± 6.6647i (3.25e+01)		
				-0.6254 ± 6.4503i (6.77e+00)	-1.0365 ± 6.3718i (3.09e+01)		
7	bips07_3078	21128,4,4	8	-0.7181 ± 5.3405i (1.92e+02)	-0.7181 ± 5.3405i (1.92e+02)	1.24	2.69
				-0.8114 ± 6.1083i (5.77e+01)	-7.5029 ± 2.2357i (7.33e+01)		
				-0.8894 ± 6.1706i (4.73e+01)	-0.8114 ± 6.1083i (5.77e+01)		
				-0.9818 ± 6.3935i (2.87e+01)	-6.0025 ± 0.0885i (4.87e+01)		
				-0.5966 ± 6.2367i (1.04e+01)	-0.8894 ± 6.1706i (4.73e+01)		
8	xingo_afonso_itaipu	13250,1,1	7	-1.2798 ± 8.0934i (4.52e−01)	-0.0153 ± 1.0916i (4.04e+00)	0.67	0.87
				-1.1021 ± 7.9841i (1.18e−01)	-1.2798 ± 8.0934i (4.52e−01)		
				-1.2713 ± 8.5866i (7.94e−02)	-14.2871 ± 5.9445i (2.03e−01)		
				-1.2180 ± 8.1776i (3.73e−02)	-0.9002 ± 7.6356i (1.36e−01)		
				-1.4127 ± 8.0192i (3.16e−02)	-0.5987 ± 0.3306i (1.31e−01)		
9	ww_vref_6405	13251,1,1	5	-0.5208 ± 2.8814i (1.45e−03)	-0.0335 ± 1.0787i (2.76e−03)	1.06	0.71
				-0.5567 ± 3.6097i (1.34e−03)	-0.5208 ± 2.8814i (1.45e−03)		
				-0.1151 ± 0.2397i (9.36e−04)	-0.5567 ± 3.6097i (1.34e−03)		
				-0.1440 (8.94e−05)	-2.9445 ± 4.8214i (1.03e−03)		
				-0.6926 ± 3.2525i (3.47e−05)	-0.1151 ± 0.2397i (9.36e−04)		
10	mimo8x8_system	13309,8,8	3	-0.3179 ± 1.0437i (7.39e−02)	-17.0961 ± 0.0919i (8.94e−02)	0.62	2.11
				-0.3464 ± 0.5796i (5.54e−02)	-0.3179 ± 1.0437i (7.39e−02)		
				-0.7168 ± 0.1238i (2.01e−02)	-0.3464 ± 0.5796i (5.54e−02)		
				-9.3974 (1.50e−02)	-16.7938 (4.48e−02)		
				-11.2079 (1.15e−02)	-0.8185 ± 0.5909i (3.05e−02)		
11	juba40k	40337,2,1	7	-0.0073 (6.10e+02)	-0.1170 ± 0.2746i (8.85e+02)	2.70	2.57
				-0.0106 (4.86e+02)	-0.0073 (6.10e+02)		
				-0.0016 (3.83e+02)	-0.0106 (4.86e+02)		
				-0.0078 (3.77e+02)	-0.0016 (3.83e+02)		
				-0.0048 (1.59e+01)	-0.0078 (3.77e+02)		

TABLE 6.4
The number of LU factorizations (# LU) and number of linear system solves (# lin sol) performed by SAMDP and Algorithm 3.1 on 11 benchmark examples are listed. Additionally, the number of restarts (# res) for SAMDP, and the subspace dimension at termination (sdim), the time for the construction of the initial subspaces in seconds (init sub) for Algorithm 3.1 are provided.

		SAMDP [28]			Alg. 3.1			
#	example	# LU	# lin sol	# res	# LU	# lin sol	sdim	init sub
1	CDplayer	25	40	0	10	80	40	0.01
2	iss	21	34	0	11	132	66	0.01
3	S40PI_n	41	64	1	15	60	30	0.04
4	S80PI_n	43	67	1	20	80	40	0.09
5	M10PI_n	53	82	2	18	216	108	0.03
6	bips07_1998	52	80	2	22	352	176	2.17
7	bips07_3078	41	64	1	23	368	184	0.92
8	xingo_afonso_itaipu	35	55	1	26	104	52	0.30
9	ww_vref_6405	61	94	2	22	88	44	0.29
10	mimo8x8_system	29	46	0	17	544	272	1.04
11	juba40k	51	79	1	24	144	48	0.92

points, while SAMDP starts with empty subspaces and gradually forms subspaces by employing exactly the same ten points. One may wonder if there is any effect of initializing the subspaces on the more robust convergence of Algorithm 3.1 to the most dominant poles.

For this reason, we modify the publicly available implementation of SAMDP so that it start with precisely the same subspaces as those employed by Algorithm 3.1. Additionally, the maximum subspace dimension for SAMDP is chosen large so that it does not restart for a fair comparison with Algorithm 3.1, which never restarts.

The results on 3 of the 11 benchmark examples are given in Table 6.5. Though not the main concern, let us first remark that when the computational cost of initialization is taken into account for SAMDP as it is done for Algorithm 3.1, SAMDP has a slower runtime than Algorithm 3.1 consistently. Secondly, a comparison of Table 6.5 with the columns of Algorithm 3.1 in Table 6.3 indicates that Algorithm 3.1 still converges to more dominant poles compared to SAMDP with initialized subspaces.

In some of the examples such as the `M10PI_n` and `ww_vref_6405` examples in Table 6.5, SAMDP now converges to dominant poles more robustly as compared to the counterpart that starts with empty subspaces. Yet, even in these examples some of the most dominant poles are missed, e.g., the fifth and fourth most dominant poles for `M10PI_n` and `ww_vref_6405` returned by Algorithm 3.1. On some other examples such as `bips07_1998` in Table 6.5, initializing subspaces seems to have negative effect on SAMDP in the sense that less dominant poles are converged.

In short, use of interpolatory projection subspaces initially may in some cases contribute. But there appears to be more important reasons for the reliable nature of Algorithm 3.1 in comparison to SAMDP. As discussed in Section 7 below, both algorithms approximate the full system with projected systems, yet the one constructed by Algorithm 3.1 seems to possess superior interpolation features.

6.3. Effect of Initial Subspace and Number of Dominant Poles Sought. Here, we illustrate how the initial subspace dimension and desired number of dominant poles affect the runtime of Algorithm 3.1 on the Brazilian power plant example `bips98_1450` with $n = 15066$, $m = p = 4$. The parameter settings are as in the previous subsection except, in one of the experiments, we vary the initial subspace dimension, keeping the number of dominant poles fixed at 5, and, in the second ex-

TABLE 6.5
SAMDP [28] with initial subspaces as those employed by Algorithm 3.1 and without any restarts.

		SAMDP with initialized subspaces, w/o restarts			
#	example	Five Most Dominant Poles	time in s	# LU	# lin sol
5	M10PI_n	1.0e+04 ·	0.33	37	163
		-0.0028 ± 0.5035i (3.99e+00)			
		-0.0032 ± 0.7530i (3.98e+00)			
		-0.0028 ± 0.8126i (2.92e+00)			
		-0.0041 ± 0.6905i (2.90e+00)			
		-0.0043 ± 1.0316i (1.12e+00)			
6	bips07_1998	-1.0365 ± 6.3718i (3.09e+01)	11.48	31	194
		-1.2743 ± 10.1195i (2.78e+01)			
		-0.7473 ± 6.2077i (1.84e+01)			
		-0.9208 ± 6.0349i (1.30e+01)			
		-0.6254 ± 6.4503i (6.77e+00)			
9	ww_vref_6405	-0.0335 ± 1.0787i (2.76e−03)	1.68	49	101
		-0.5208 ± 2.8814i (1.45e−03)			
		-0.5567 ± 3.6097i (1.34e−03)			
		-0.1151 ± 0.2397i (9.36e−04)			
		-1.4595 ± 10.7705i (3.63e−04)			

periment, the number of dominant poles is varied while keeping the number of initial interpolation points fixed at 15.

The results of the first experiment are displayed in the left-hand plot in Figure 6.2. The initial subspace dimension is increased by the way of increasing the number of initial interpolation points. As usual, the initial interpolation points are selected as the dominant poles of a reduced system obtained from an application of the algorithm in [1]. Initially, when number of initial interpolation points is about 8-12, the runtime does not change by much, but then it increases more or less monotonically. The reason is that the number of subspace iterations is already small at about 3-4 with the number of initial interpolations about 8-12. For larger number of interpolation points, the number of subspace iterations still remains about 3-4, but there is the additional cost of interpolating at further points in the form of computing further LU factorizations and solving triangular systems. This is a behavior we typically observe on the benchmark examples. A small number of initial interpolation points is usually sufficient for accuracy at a reasonable computational burden.

In the second experiment whose results are shown on the right-hand side in Figure 6.2, the runtime of Algorithm 3.1 usually increases as the number of dominant poles sought is increased. Only in this example we set the termination tolerance on the residuals as $\texttt{tol} = 10^{-6}$ to avoid convergence difficulties for larger number of dominant poles. The increase in the runtime is in harmony with the increase in the number of LU factorizations performed, also depicted in the plot. The plot of the number of triangular systems solved, the other important factor affecting the runtime, with respect to the number of dominant poles is omitted, as it resembles the one for LU factorizations scaled up by about a factor of 16. This behavior of runtime as a function of number of dominant poles is expected in general, but how quickly the runtime and number of LU factorizations grow vary from example to example. In this particular example, the runtime of SAMDP grows faster than that of Algorithm 3.1 with respect to the number of dominant poles.

6.4. Modal Model Reduction on `bips98_1450`. An application of modal model reduction, in particular an application of (1.6), to `bips98_1450` using the 20 poles computed by Algorithm 3.1, their complex conjugates and corresponding eigenvectors gives rise to a reduced model $H_{\mathsf{red}}(s)$ of order 40. The largest and smallest

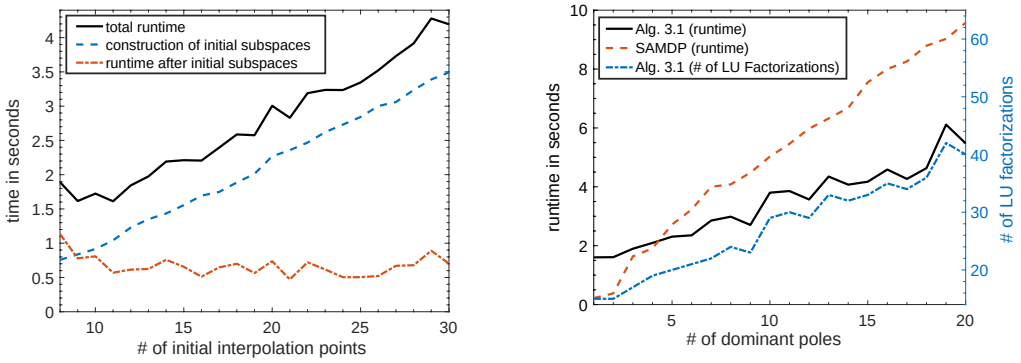


FIG. 6.2. *The performance on the* `bips98_1450` *example as a function of number of initial interpolation points (left), and as a function of the number of dominant poles sought (right).*

singular values of $H_{\mathsf{red}}(s)$ along with $H(s)$ restricted to the imaginary axis are depicted on the left in Figure 6.3. In the same figure on the right, the error of this reduced model is compared with the reduced model of order 40 obtained similarly but by using the 20 most dominant poles and conjugates returned by SAMDP instead. The poles returned by Algorithm 3.1 has higher dominance metrics. As a result they lead to a reduced model with smaller error. Note that once again we set the termination tolerance as $\texttt{tol} = 10^{-6}$ for Algorithm 3.1 and SAMDP in these experiments.

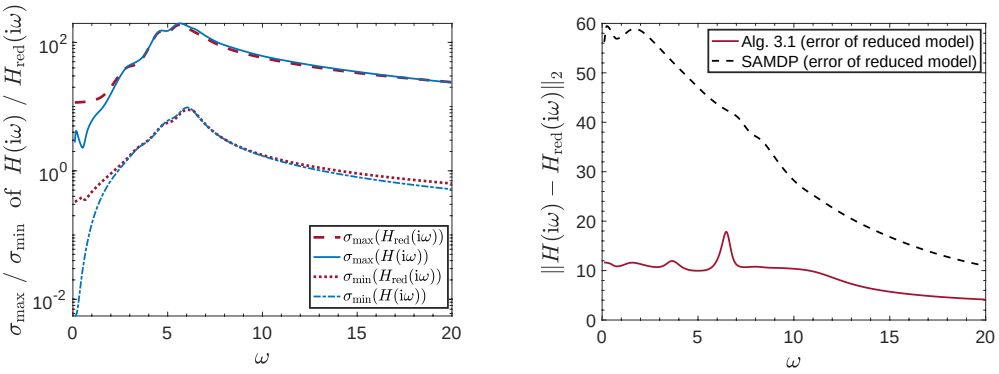


FIG. 6.3. *Applications of modal model reduction to* `bips98_1450` *so that* $H_{\mathrm{red}}(s)$ *is as in (1.6). (Left) 20 most dominant poles and complex conjugates from Algorithm 3.1 are used to construct* $H_{\mathrm{red}}(s)$*. Largest, smallest singular values* $\sigma_{\max}$*,* $\sigma_{\min}$ *of* $H(\mathrm{i}\omega)$*,* $H_{\mathrm{red}}(\mathrm{i}\omega)$ *are plotted. (Right) 20 most dominant poles and complex conjugates from both algorithms lead to two different* $H_{\mathrm{red}}(s)$*. Error* $\|H(\mathrm{i}\omega) - H_{\mathrm{red}}(\mathrm{i}\omega)\|_2$ *is plotted for each* $H_{\mathrm{red}}(\mathrm{i}\omega)$*.*

7. Interpolatory Properties of SAMDP vs. Algorithm 3.1. The original dominant pole algorithm (DPA) [22] and its variant for multiple-input-multiple-output systems [23] are Newton iterations on eigenvalue functions. They or their minor modifications can be viewed as Newton's methods for finding the zeros of extensions of $1/\|H(s)\|_2$, or $1/\lambda_{\max}(H(s))$ in case the number of inputs and outputs are equal (i.e., in case $m = p$), that are defined and smooth at the poles of $H(s) = C(sE - A)^{-1}B$, where $\lambda_{\max}(H(s))$ denotes the largest eigenvalue of $H(s)$ in modulus.

The subspace acceleration is one of the components introduced later to deal with multiple poles that resulted in the subspace accelerated multiple-input-multiple-output dominant pole algorithm (SAMDP) [28]. Together with subspace acceleration, Newton scheme can be fully abandoned. Indeed, the available implementation of SAMDP with default settings that we make use of do not employ Newton's method. Instead, just like Algorithm 3.1, it also forms left and right projection subspaces, and computes the most dominant pole τ of the projected system. Then it expands the right and left projection subspaces, say to $\mathcal{V}$ and $\mathcal{W}$, with the inclusion of directions $(A-\tau E)^{-1}Bu$ and $(A-\tau E)^{-*}Cv$, respectively, where u and v denote left and right singular vectors corresponding to the largest singular value $\sigma_{\max}(H(\tau))$ of $H(\tau)$, or alternatively the left and right eigenvectors corresponding to $\lambda_{\max}(H(\tau))$ if $m=p$. These inclusions in terms of the projected system $H_{\rm red}^{\mathcal{W},\mathcal{V}}(s)$ result in[2]

$$H(\tau)u \;=\; H_{\rm red}^{\mathcal{W},\mathcal{V}}(\tau)u, \quad v^*H(\tau) \;=\; v^*H_{\rm red}^{\mathcal{W},\mathcal{V}}(\tau), \quad v^*H'(\tau)u \;=\; v^*[H_{\rm red}^{\mathcal{W},\mathcal{V}}]'(\tau)u.$$

If u and v are singular vectors corresponding to $\sigma_{\max}(H(\tau))$, the first two interpolation properties guarantee $\|H(\tau)\|_2 \le \|H_{\rm red}^{\mathcal{W},\mathcal{V}}(\tau)\|_2$, but not necessarily the equality is attained. In case, the equality is attained, then the third interpolation property implies $\frac{d}{ds}\|H(s)\|_2\big|_{s=\tau} = \frac{d}{ds}\|H_{\rm red}^{\mathcal{W},\mathcal{V}}(s)\|_2\Big|_{s=\tau}$. On the other hand, if u and v are eigenvectors corresponding to $\lambda_{\max}(H(\tau))$, and they happen to be also left and right eigenvectors corresponding to $\lambda_{\max}(H_{\rm red}^{\mathcal{W},\mathcal{V}}(\tau))$, then $\frac{d}{ds}\lambda_{\max}(H(s))\big|_{s=\tau} = \frac{d}{ds}\lambda_{\max}(H_{\rm red}^{\mathcal{W},\mathcal{V}}(s))\big|_{s=\tau}$. The reduced problems by SAMDP does not possess any interpolatory properties for the second and higher-order derivatives.

In contrast, Algorithm 3.1 when $m=p$ for a dominant pole estimate τ expands the subspaces, say into $\mathcal{V}$ and $\mathcal{W}$, with the inclusion of $\mathrm{Ran}[(A-\tau E)^{-1}B]$ and $\mathrm{Ran}[(A-\tau E)^{-*}C]$ along with other directions (similarly when $m \ne p$). These inclusions always guarantee $H(\tau) = H_{\rm red}^{\mathcal{W},\mathcal{V}}(\tau)$ and $H'(\tau) = [H_{\rm red}^{\mathcal{W},\mathcal{V}}]'(\tau)$, properties stronger than $\|H(\tau)\|_2 = \|H_{\rm red}^{\mathcal{W},\mathcal{V}}(\tau)\|_2$, $\frac{d}{ds}\|H(s)\|_2\big|_{s=\tau} = \frac{d}{ds}\|H_{\rm red}^{\mathcal{W},\mathcal{V}}(s)\|_2\big|_{s=\tau}$, $\frac{d}{ds}\lambda_{\max}(H(s))\big|_{s=\tau} = \frac{d}{ds}\lambda_{\max}(H_{\rm red}^{\mathcal{W},\mathcal{V}}(s))\big|_{s=\tau}$. The reduced problems by Algorithm 3.1 interpolate even the second and third derivatives, and possibly higher-order derivatives of the full problem at τ according to Lemma 3.1.

Table 7.1 summarizes the interpolatory properties of SAMDP and Algorithm 3.1, when $m=p$ and u, v are left, right singular vectors corresponding to $\sigma_{\max}(H(\tau))$ in the case of SAMDP. Clearly, Algorithm 3.1 has better interpolation properties. As a result, the reduced function $H_{\rm red}^{\mathcal{W},\mathcal{V}}(s)$ constructed by Algorithm 3.1 is a more accurate global approximation for $H(s)$ in comparison to the reduced function by SAMDP. This theoretical fact supports the more robust convergence of Algorithm 3.1 to dominant poles compared to SAMDP observed numerically in Section 6.2.

8. Software. A Matlab implementation of Algorithm 3.1 is publicly available on the web at https://zenodo.org/record/5103430.

This implementation can be run on the benchmark examples in Section 6.2 using the script `demo_on_benchmarks`. To estimate `k` dominant poles of a descriptor system `(A,E,B,C,D)`, it can simply be called as `[evals,evecs]=DP_main(A,E,B,C,k)`, which would return the computed poles in `evals`, and corresponding eigenvectors in `evecs`.

9. Concluding Remarks. The dominant poles of the transfer function of a descriptor system is used in the literature for model order reduction. Additionally,

[2]We are not sure if the authors were aware of these interpolation properties at the time [28] was published. The interpolation results have been put into use mostly after 2005.

TABLE 7.1
A Comparison of the Interpolatory Properties of Algorithm 3.1 and SAMDP.

	SAMDP	Algorithm 3.1
Right Subspaces	$(A-\tau E)^{-1}Bu$, u is left singular vec. cor. $\sigma_{\max}(H(\tau))$	$\text{Ran}[\{(A-\tau E)^{-1}E\}^j (A-\tau E)^{-1}B]$ for $j=0,1,\ldots,q$
Left Subspaces	$(A-\tau E)^{-*}Cv$, v is right singular vec. cor. $\sigma_{\max}(H(\tau))$	$\text{Ran}[(C(A-\tau E)^{-1}\{E(A-\tau E)^{-1}\}^j)^*]$ for $j=0,1,\ldots,q$
Transfer Functions	$\|H(\tau)\|_2 \leq \|H_{\text{red}}^{\mathcal{W},\mathcal{V}}(\tau)\|_2$	$H(\tau)=H_{\text{red}}^{\mathcal{W},\mathcal{V}}(\tau)$
First Derivatives	$\frac{d}{ds}\|H(s)\|_2\big\|_{s=\tau} = \frac{d}{ds}\|H_{\text{red}}^{\mathcal{W},\mathcal{V}}(s)\|_2\big\|_{s=\tau}$ if $\|H(\tau)\|_2 = \|H_{\text{red}}^{\mathcal{W},\mathcal{V}}(\tau)\|_2$	$H'(\tau)=[H_{\text{red}}^{\mathcal{W},\mathcal{V}}]'(\tau)$
Higher Derivatives	No apparent interpolatory properties	$H^{(j)}(\tau)=[H_{\text{red}}^{\mathcal{W},\mathcal{V}}]^{(j)}(\tau)$ for $j=2,3,\ldots,2q+1$

dominant poles provide information about how the transfer function behaves along the imaginary axis, in particular about regions on the imaginary axis where the transfer function attains large norm. Hence, it is plausible to initialize the algorithms for large-scale $\mathcal{L}_\infty$-norm computation based on information retrieved from dominant poles.

Here, we have proposed an interpolatory subspace framework to estimate a prescribed number of dominant poles for a descriptor system. At every iteration, the dominant poles of a projected small problem is computed using standard eigenvalue solvers such as the ones based on the QZ algorithm. Then the projection subspaces are expanded so that the transfer function of the projected problem after expansion Hermite interpolates the original transfer function at these computed dominant poles. We have shown that the proposed framework converges at least at a quadratic rate under mild assumptions, and verified this result on real benchmark examples. Our numerical experiments indicate that on benchmark examples the framework locates the dominant poles more reliably in comparison to SAMDP [28], one of the existing methods for dominant pole estimation.

It may be possible to extend the framework introduced here to more general class of transfer functions beyond rational functions, such as the transfer functions associated with delay systems. Moreover, a careful incorporation of the framework here for dominant pole estimation to initialize the algorithms for large-scale $\mathcal{L}_\infty$-norm computation may be an important step. It may pave the way for accurate and efficient computation of $\mathcal{L}_\infty$ norm for descriptor systems of large order.

Acknowledgements. The author is grateful to two anonymous referees who provided invaluable feedback on the initial versions of this manuscript.

REFERENCES

[1] A. ALIYEV, P. BENNER, E. MENGI, P. SCHWERDTNER, AND M. VOIGT, *Large-scale computation of $\mathcal{L}_\infty$-norms by a greedy subspace method*, SIAM J. Matrix Anal. Appl., 38 (2017), pp. 1496–1516.

[2] A. ALIYEV, P. BENNER, E. MENGI, AND M. VOIGT, *A subspace framework for $\mathcal{H}_\infty$ norm minimization*, SIAM J. Matrix Anal. Appl., 41 (2020), pp. 928–956.

[3] A. ALIYEV, V. MEHRMANN, AND E. MENGI, *Approximation of stability radii for large-scale dissipative hamiltonian systems*, Adv. Comput. Math., 46 (2020).

[4] A. C. ANTOULAS, *Approximation of Large-Scale Dynamical Systems*, vol. 6 of Adv. Des. Control, SIAM Publications, Philadelphia, PA, 2005.

[5] A. C. Antoulas, C. A. Beattie, and G. S., *Interpolatory model reduction of large-scale dynamical systems*, Springer-Verlag, 2010, pp. 3–58.
[6] R. Aziz, E. Mengi, and M. Voigt, *Derivative interpolating subspace frameworks for nonlinear eigenvalue problems*. arXiv preprint arXiv:2006.14189 [math.NA], 2020. Submitted.
[7] C. Beattie and S. Gugercin, *Interpolatory projection methods for structure-preserving model reduction*, Systems Control Lett., 58 (2009), pp. 225–232.
[8] P. Benner and T. Mitchell, *Faster and more accurate computation of the H_∞ norm via optimization*, SIAM J. Sci. Comput., 40 (2018), pp. A3609–A3635.
[9] P. Benner and M. Voigt, *A structured pseudospectral method for $\mathcal{H}_\infty$-norm computation of large-scale descriptor systems*, Math. Control Signals Systems, 26 (2014), pp. 303–338.
[10] S. Boyd and V. Balakrishnan, *A regularity result for the singular values of a transfer matrix and a quadratically convergent algorithm for computing its L_∞-norm*, Systems Control Lett., 15 (1990), pp. 1–7.
[11] N. A. Bruinsma and M. Steinbuch, *A fast algorithm to compute the H_∞-norm of a transfer function matrix*, Systems Control Lett., 14 (1990), pp. 287–293.
[12] M. A. Freitag, A. Spence, and P. Van Dooren, *Calculating the H_∞-norm using the implicit determinant method*, Linear Algebra Appl., 35 (2014), pp. 619–635.
[13] K. Gallivan, A. Vandendorpe, and P. Van Dooren, *Model reduction of mimo systems via tangential interpolation*, SIAM J. Matrix Anal. Appl., 26 (2004), pp. 328–349.
[14] F. R. Gantmacher, *The Theory of Matrices*, vol. 1, Chelsea, 1959.
[15] N. Gräbner, V. Mehrmann, S. Quraishi, C. Schröder, and U. von Wagner, *Numerical methods for parametric model reduction in the simulation of disc brake squeal*, Z. Angew. Math. Mech., 96 (2016), pp. 1388–1405.
[16] S. Gugercin, A. C. Antoulas, and C. Beattie, *$\langle_2$ model reduction for large-scale linear dynamical systems*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 609–638.
[17] S. Gugercin, T. Stykel, and S. Wyatt, *Model reduction of descriptor systems by interpolatory projection methods*, SIAM J. Sci. Comput., 35 (2013), pp. B1010–B1033.
[18] N. Guglielmi, M. Gürbüzbalaban, and M. L. Overton, *Fast approximation of the H_∞-norm via optimization over spectral value sets*, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 709–737.
[19] D. Hinrichsen and A. Pritchard, *Mathematical Systems Theory I: Modelling, State Space Analysis, Stability and Robustness*, Texts in Applied Mathematics, Springer Berlin Heidelberg, 2011.
[20] D. Hinrichsen and A. J. Pritchard, *Stability radius for structured perturbations and the algebraic Riccati equation*, Systems Control Lett., 8 (1986), pp. 105–113.
[21] N. Martins, *The dominant pole spectrum eigensolver [for power system stability analysis]*, IEEE Transactions on Power Systems, 12 (1997), pp. 245–254.
[22] N. Martins, L. T. G. Lima, and H. J. C. P. Pinto, *Computing dominant poles of power system transfer functions*, IEEE Transactions on Power Systems, 11 (1996), pp. 162–170.
[23] N. Martins and P. E. M. Quintao, *Computing dominant poles of power system multivariable transfer functions*, IEEE Transactions on Power Systems, 18 (2003), pp. 152–159.
[24] C. Mehl, V. Mehrmann, and P. Sharma, *Stability radii for linear Hamiltonian systems with dissipation under structure-preserving perturbations*, SIAM J. Matrix Anal. Appl., 37 (2016), pp. 1625–1654.
[25] T. Mitchell and M. L. Overton, *Hybrid expansion-contraction: a robust scaleable method for approximating the H_∞ norm*, IMA J. Numer. Anal., 36 (2016), pp. 985–1014.
[26] Y. Nakatsukasa, O. Sète, and L. N. Trefethen, *The AAA algorithm for rational approximation*, SIAM J. Sci. Comput., 40 (2018), pp. A1494–A1522.
[27] J. Rommes, *Modal Approximation and Computation of Dominant Poles*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 177–193.
[28] J. Rommes and N. Martins, *Efficient computation of multivariable transfer function dominant poles using subspace acceleration*, IEEE Trans. Power Syst., 21 (2006), pp. 1471–1483.
[29] J. Rommes and N. Martins, *Efficient computation of transfer function dominant poles using subspace acceleration*, IEEE Trans. Power Syst., 21 (2006), pp. 1218–1226.
[30] J. Rommes and G. L. G. Sleijpen, *Convergence of the dominant pole algorithm and Rayleigh quotient iteration*, SIAM J. Matrix Anal. Appl., 30 (2008), pp. 346–363.
[31] A. J. van der Schaft and D. Jeltsema, *Port-Hamiltonian systems theory: An introductory overview*, Foundations and Trends in Systems and Control, 1 (2014), pp. 173–378.
[32] D. Vizer, G. Mercère, O. Prot, and E. Laroche, *H_∞-norm-based optimization for the identification of gray-box LTI state-space model parameters*, Systems Control Lett., 92 (2016), pp. 34–41.