Peter Benner Patrick Kürschner Jens Saak

A Goal-Oriented Dual LRCF-ADI for Balanced Truncation

Max Planck Institute Magdeburg Preprints

MPIMD/12-01 **January 5, 2012**

Impressum:

Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg

Publisher:
Max Planck Institute for Dynamics of Complex Technical Systems

Address:
Max Planck Institute for Dynamics of Complex Technical Systems
Sandtorstr. 1
39106 Magdeburg

www.mpi-magdeburg.mpg.de/preprints

Max Planck Institute Magdeburg Preprints

Peter Benner Patrick Kürschner Jens Saak

A Goal-Oriented Dual LRCF-ADI for Balanced Truncation

MPIMD/12-01 January 5, 2012

Abstract

In this contribution we propose a novel dual Lyapunov solver that computes low rank factors of the reachability and observability Gramians of a control system simultaneously. This is especially helpful in balanced truncation model order reduction applications, where the singular values of the product of these Gramian factors are the basis of the truncation process. For stopping the dual iteration scheme we therefore propose a tailored stopping criterion aiming at the accurate computation of these singular values.

Contents

Author's addresses:

Peter Benner / Patrick Kürschner / Jens Saak
Computational Methods in Systems and Control Theory, Max Planck Institute for Dynamics of Complex Technical Systems,
Sandtorstr. 1,
39106 Magdeburg, Germany,
({benner,kuerschner,saak}@mpi-magdeburg.mpg.de)

1. Introduction

Balanced truncation (BT) introduced by [11] has shown to be a reliable method for model order reduction (MOR) (see, e.g., [2]). The basic idea is to compute the reachability and observability Gramians of a linear system from which information on the dominant states to be preserved in the reduced order model (ROM) can be extracted. It is especially attractive when the reduced order model has to be of a certain quality since it features a computable a priori error bound. This is especially true for small to medium size systems. There the dense storage of the system is possible and computation of the two system Gramians is still feasible using classical direct methods with $\mathcal{O}(n^3)$ complexity. For large and very large systems that can only be stored in sparse or data sparse formats, the Gramians are usually approximated by low rank or data sparse representations. Here we focus on the sparse case and thus on low rank representations of the Gramians. There are several methods in the literature for computing the low rank Gramians as the factorizations of the solutions of the reachability and observability Lyapunov equations. The low rank Cholesky factor alternating directions implicit (LRCF-ADI) iteration (see, e.g., [10, 4]), is one of the most efficient and most deeply investigated among these methods.

The most crucial question in iterative methods in general is when to stop the iteration and accept the accuracy of the current iterate. In the special case at hand this is even more crucial. The residual, whose smallness is often used as a stopping criterion in iterative methods, is a dense square matrix and thus not computable itself. Norms of the residual can however be computed or approximated without forming the residual explicitly. Still these computations easily become as expensive as the actual iteration step. To make it even worse, observations in practical examples show that the two Lyapunov residuals do not directly relate to the accuracy of the reduced order model.

In this contribution we therefore propose a more goal oriented stopping criterion. In the case of balanced truncation the actual properties of interest are the Hankel singular values of the system since these are applied as the basis of the truncation. We develop a dual ADI iteration that solves the two dual Lyapunov equations simultaneously and stops as soon as the Hankel singular values are found to the desired accuracy.

2. Balanced Truncation for Generalized Linear Time Invariant Systems

Consider the system

$$E\dot{x}(t) = Ax(t) + Bu(t), \tag{1}$$

$$y(t) = Cx(t), \tag{2}$$

where $E, A \in \mathbb{R}^{n\times n}$, E is (for ease of presentation) assumed to be invertible and the pencil has only stable eigenvalues, i.e., the system is asymptotically stable.

In BT-MOR the main task is to solve the generalized reachability and observability Lyapunov equations

$$APE^T + EPA^T = -BB^T, \tag{3}$$

$$A^TQE + E^TQA = -C^TC. \tag{4}$$

As the system is assumed to be stable and thus P and Q are positive (semi-)definite, there exist Cholesky(-like) factorizations $P = S^TS$ and $Q = R^TR$. In the so-called *square-root* balanced truncation (SRBT) algorithms introduced by [13, 9], these are used to define the projection matrices

$$T_l := \Sigma_1^{-\frac{1}{2}} U_1^T R^T \tag{5}$$

$$\text{and } T_r := SV_1\Sigma_1^{-\frac{1}{2}} \tag{6}$$

determining the reduced order model as

$$\hat{E}\dot{\hat{x}}(t) = \hat{A}\hat{x}(t) + \hat{B}u(t), \tag{7}$$

$$\hat{y}(t) = \hat{C}\hat{x}(t), \tag{8}$$

with

$$\hat{E} := T_lET_r, \quad \hat{A} := T_lAT_r, \quad \hat{B} := T_lB \quad \text{and} \quad \hat{C} := CT_r.$$

The matrices $\Sigma_1^{-\frac{1}{2}}$, U_1 and V_1 in equations (5), (6) are determined via the singular value decomposition

$$R^TES = U\Sigma V^T = \begin{bmatrix} U_1 U_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix}, \tag{9}$$

with decreasingly ordered singular values – the Hankel singular values of the system.

Note that in case E is symmetric positive definite (e.g. a finite element mass matrix) we compute the transformation matrices T_l, T_r with respect to the inner product induced by E. In fact by construction of the transformation we always get $\hat{E} = T_lET_r = I$ if E is invertible.

3. Cross-Gramian and Approximate Invariant Subspace Approaches

The computation of the Gramian factors in (9) is a computationally expensive and challenging task. In this section we will review two possible approaches to reduce the computational complexity discussed in the literature. The Cross-Gramian approach is especially useful in the case of a symmetric system, i.e., $A = A^T$. There it can be shown that for the solution X of the equation

$$\tilde{A}X + X\tilde{A} = -\tilde{B}\tilde{C}, \tag{10}$$

one has $X^2 = \tilde{P}\tilde{Q}$. Here $\tilde{A} := E_C^{-1} A E_C^{-T}$, $\tilde{B} := E_C^{-1} B$ and $\tilde{C} := C E_C^{-T}$ for E_C a Cholesky factor of E. In turn for the Hankel singular values we find $\sigma_i = |\lambda_i(X)|$ (see, e.g., [5, 6]). Thus, replacing the SVD in (9) by the computation of bi-orthogonal bases of the dominant eigenspaces of X, one can proceed as above (cf. [1]). The obvious advantage is that only one matrix equation needs to be solved. On the other hand we have a very restricted applicability. The number of inputs and outputs to the system have to coincide and the system needs to be symmetric, or a symmetrizer has to exist which is usually non trivial to find, especially numerically.

A second approach that tries to circumvent the necessity for a symmetrizer, but keeps the idea of approximating the dominant eigenspaces is due to [14]. The approximate implicit subspace iteration with alternating directions (AISIAD) successively computes invariant subspaces of P and Q and uses the subspace for P to reduce the amount of computations in the next step for Q and vice versa. The key ingredient in the efficiency improvement is to transform the two Lyapunov equations in (3), (4) into specially structured Sylvester equations. Here we show this for equation (3)

$$APV_i + PV_iV_i^T A^T V_i + M_i + BB^T V_i = 0. \tag{11}$$

The coefficient matrix in the first term stays untouched, whereas the one in the second term is transformed into a small and dense matrix $H_i^T := V_i^T A^T V_i$. The matrix $M_i := P(I - V_iV_i^T)A^T V_i$ needs to be approximated efficiently in this framework. These resulting equations of the form

$$AX + XH + M = 0$$

can then be solved by an implicitly-restarted block Arnoldi iteration as in [14], or using a combination of sparse direct solvers for A and dense techniques for the small and dense matrix H following [3].

Our approach is more closely related to the balanced truncation idea. We stick with the computation of the Gramian factors but replace them by low rank factors and exploit that the two Lyapunov equations are dual to each other to decrease the amount of computations taken. This approach will be presented in the following section.

4. Low Rank Gramian Factors and the Dual ADI Idea

In the case of large and sparse systems with $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$ and $m, p \ll n$, computing the triangular Cholesky factors $S \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{n\times n}$ is infeasible due to memory and complexity limitations. They are replaced by low rank Cholesky factors, i.e., $S \in \mathbb{R}^{n\times k_S}$ and $R \in \mathbb{R}^{n\times k_R}$ exploiting the usually very low (numerical) rank of the Gramians in this case.

4.1. Low Rank ADI for the Gramian Factors

The factors are computed by a low rank Lyapunov equation solver. We are here focusing on the variant of LRCF-ADI (e.g., [12]) successively computing S as in

$$\begin{aligned} V_1 &= \alpha_1 \left(A + p_1 E\right)^{-1} B, \\ S_1 &= V_1, \\ V_i &= \alpha_i \left[I - (p_i + \overline{p_{i-1}}) \left(A + p_i E\right)^{-1}\right] E V_{i-1}, \\ S_i &= \left[S_{i-1}, V_i\right]. \end{aligned} \tag{12}$$

For R this translates to

$$\begin{aligned} W_1 &= \alpha_1 \left(A^T + p_1 E^T\right)^{-1} C, \\ R_1 &= W_1, \\ W_i &= \alpha_i \left[I - (p_i + \overline{p_{i-1}}) \left(A^T + p_i E^T\right)^{-1}\right] E^T W_{i-1}, \\ R_i &= \left[S_{i-1}, W_i\right]. \end{aligned} \tag{13}$$

Here $\alpha_1 = \sqrt{-2\,\mathrm{Re}\,(p_1)}$ and $\alpha_i = \sqrt{\frac{\mathrm{Re}\,(p_i)}{\mathrm{Re}\,(p_{i-1})}}$ for $i \geq 2$. When applying sparse direct solvers to the linear system solves, the LU decomposition

$$(A + p_k E) = L_k U_k, \tag{14}$$

is the by far most time consuming step.

4.2. Efficient implementation of the dual ADI iteration

Now rewriting the inverses in (13) in the form

$$(A + p_k E)^{-1} = (A + \overline{p_k} E)^{-T},$$

we observe that the LU factorization (14) can be reused in the triangular solves for the dual equation. On the first glimpse the $\overline{p_k}$ here is a problem. However if we keep in mind that a proper set of shift parameters consists of pairs of complex conjugate shifts applied one after the other, we immediately see that reusing (14) only switches the order of the complex pairs of shifts in the dual equation. So we can do the cheap forward-backward-solves with the same triangular factors as in (13). Thus we can efficiently formulate a dual iteration simultaneously computing both low rank Gramian factors as presented in Algorithm 1.

4.3. Stopping the Dual Iteration

Usually criteria like the normalized residual

$$\mathrm{res}_k(S_k) = \gamma^{-1} \| A S_k S_k^T E^T + E S_k S_k^T A^T + B B^T \|_2, \tag{15}$$

Algorithm 1 The dual low rank ADI iteration

Input: E, A, B, C from (1) and a proper set of shifts $p_k \in \mathbb{C}^-$, $k = 1, \ldots, \texttt{maxiter}$
Output: S, R ensuring good approximation of Σ_1 in (9).

$[L, U] = \texttt{lu}(A + p_1 E)$
$V_1 = \sqrt{-2\,\mathrm{Re}\,(p_1)}U^{-1}L^{-1}B$
$W_1 = \alpha_1 L^{-T}U^{-T}C^T$
$S_1 = V_1$
$R_1 = W_1$
for $i = 2, \ldots, \texttt{maxiter}$ **do**
  $[L, U] = \texttt{lu}(A + p_i E)$
  $\alpha_i = \sqrt{\frac{\mathrm{Re}\,(p_i)}{\mathrm{Re}\,(p_{i-1})}}$
  $V_i = \alpha_i \left(V_{i-1} - (p_i + \overline{p_{i-1}})U^{-1}L^{-1}(EV_{i-1})\right)$
  $W_i = \alpha_i \left(W_{i-1} - (p_i + \overline{p_{i-1}})L^{-T}U^{-T}(E^T W_{i-1})\right)$
  $S_i = [S_{i-1}, V_i]$
  $R_i = [R_{i-1}, W_i]$
  if converged **then**
    STOP
  end if
end for

with $\gamma = \|BB^T\|_2$ or $\gamma = \|BB^T\|_2 + \|E\|_2\|A\|_2\|Z\|_2^2$, or the relative change of the factor

$$\mathrm{rc}(S_k) = \frac{\|V_k\|_F}{\|S_k\|_F}, \tag{16}$$

are employed to stop the iterations in (12) and (13). While the relative change can cheaply be accumulated, the residual is usually expensive to compute. In many examples the evaluation of the stopping criterion is therefore almost as expensive as the actual iteration step.

Besides that, the residual of the Lyapunov equations often is totally unrelated to the accuracy of the reduced order model that we are actually interested in. That means in practice we have observed very good reduced order model accuracies, even when the ADI did not converge at all. On the other hand we have seen examples where one of the ADI iterations converged very much faster than the other one. Since the upper limit to the reduced order model is the smaller of the two column dimensions of S and R, this can limit the accuracy of the reduced order model and prevent us from fulfilling prescribed error bounds.

Our novel idea to overcome these issues in the dual ADI process is to completely skip classical stopping criteria and use something more problem oriented. The actual properties of interest in the square root method for balanced truncation are the Hankel singular values of the original system. To be precise we want to capture the leading Hankel singular values as accurate as possible since these describe the dominant dynamics of the system and thus need to be reflected in the reduced order model.

Let us for the time being assume we are searching for a fixed order (say $r \in \mathbb{N}$)

reduced order model. The two Gramian factors can be employed to compute (9) during the iteration since we compute them simultaneously anyway. Now we can monitor the change of the leading r singular values in Σ_1. Once these singular values stagnate we have matched the corresponding subsystem and thus get a good reduced order model evaluating (7). This way we find a reliable criterion when to stop the iteration in contrast to the residual that may not tell us anything about the approximation. Note that in finite arithmetic it is sufficient to drive the relative change of the leading Hankel singular values below a certain tolerance. Here relative is to be understood as relative to the largest singular value. For example, for the tolerance equal to machine precision it does not make sense to compute the small singular values more exactly since due to rounding the contribution of the further digits will not be seen in the results anyway.

To increase the efficiency of the implementation even further we can use a start up phase in which we do not evaluate the stopping criterion at all. This phase is predeterminated by the size of the matrices B and C. It should at least be running as long as S and R do not have a minimum of r linearly independent columns each.

4.4. Convergence of the Hankel Singular Values

The quantitative analysis of the convergence of the Hankel singular values is still under investigation. However here we want to report the basic qualitative ideas to motivate the stopping criterion above. From the practitioners point of view it is obvious that the criterion is the right thing to look at if the Hankel singular values converge. Now we want to show that at least we can expect them to do so and give a qualitative explanation for the observation that the leading ones converge from below to the actual Hankel singular values of the original system while we increase the dimension of the subspaces spanned by the columns of R and S. For ease of representation we assume $E = I$ in the following. From (9) we know that the Hankel singular values correspond to the singular values of $M := SR^T$. Clearly we have that

$$\sigma_i(M) = \sqrt{\lambda_i(M^T M)} = \sqrt{\lambda_i(PQ)},$$

for $i = 1, \ldots, n$, i.e., computing the Hankel singular values is equivalent to the symmetric eigenvalue problem for $M^T M$. Applying a Krylov subspace method to the symmetric eigenvalue problem we know that the Ritz values converge from the inside of the spectrum to the eigenvalues, since they are found as Rayleigh quotients that have to lie between the real eigenvalues due to their interlacing property ([8]). Now noting that the ADI iteration in fact forms rational Krylov subspaces (see, e.g., [10]) with respect to A and B, or A^T and C^T, forming the columns of the Gramian factors it is clear that the leading Hankel singular values are approximated from below, whereas the smallest ones converge from above.

4.5. Acceleration of the Algorithm

If the matrices forming the system are badly conditioned the iteration may produce Gramian factors whose numerical rank is smaller than the actual number of columns

due to rounding errors. Then the SVDs computed in the evaluation of the stopping criterion become considerably larger than required, as well. Here the column compression proposed, e.g., in [12] can help to further reduce the execution time. The SVD for the matrix SR^T is a dense $\mathcal{O}(\tilde{n}^3)$ operation, where $\tilde{n}$ is the leading dimension of the Gramian factor product in (9), i.e., the larger of the two column numbers. Now applying column compression to R and S such that the resulting compressed factors have a number of columns equal to their ranks we can minimize $\tilde{n}$ which can give a strong runtime reduction due to the cubic effort in the SVD. Note that this will only pay off if the number of columns is considerably larger than the actual numerical rank. Otherwise the additional work for the compression may not be compensated by the gain from the reduction of $\tilde{n}$.

5. Numerical Experiments

The numerical experiments reported in this section have been generated on a Dell Precision Workstation with:

- CPU: 2x Intel® Xeon® W3503 @ 2.40GHz
- Cache: 4MB
- RAM: 6GB
- OS: Ubuntu Linux 10.04LTS
- 32bit kernel 2.6.32-25-generic-pae
- MATLAB® 7.11.0 (R2010b)

Note that MATLAB in this setting can only use 2GB of the main memory due to the 32bit limit.

We investigate two test examples. The first is the Semi-discretized Heat Transfer Problem for Optimal Cooling of Steel Profiles from the Oberwolfach benchmark collection for model reduction[1]. To be able to compute the Hankel singular values with classical dense methods we use the version with $n = 1357$ degrees of freedom. The model has 7 inputs and 6 outputs and is of the form (1). We will in the following shortly use *rail model* to refer to this example.

The other example requires some additional implementation work. We consider the BIPS07_1693[2] model introduced in [7]. This model is of the form (1) as well, but with an E matrix that is singular. Thus we have to deal with a differential algebraic equation in this case. Fortunately the block in A corresponding to the nullspace of E is theoretically invertible and thus the model is of index 1. In [7] the authors showed how the low rank ADI can be extended to such equations efficiently. Since the SVD in (9) stays essentially the same, the stopping criterion proposed in Section 4.3 can

[1] `http://simulation.uni-freiburg.de/benchmark`
[2] `http://sites.google.com/site/rommes/software`

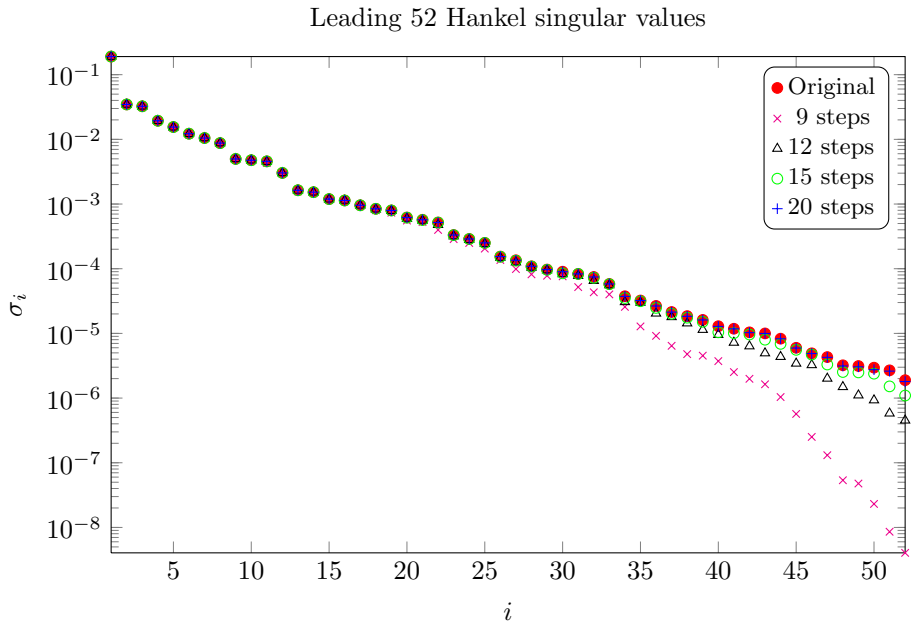

Figure 1: Hankel singular values for several low rank Gramian approximations after certain numbers of dual ADI iteration steps.

directly be applied there, too. Also the extension of the dual ADI formulation is straight forward. The model has 4 inputs and 4 outputs. We have explicitly stated that the A_{22} block is theoretically invertible since it shows to have a gigantic condition number. We chose this example to show the robustness of the Hankel singular value convergence to such bad conditioning of the model. The model is of order 13 275 but the Schur complement formulation erasing the algebraic constraints is of order 1693 only.

Both examples are of a size that allows the computation of the Hankel singular values by a classical dense approach. We will refer to the singular values resulting from these computations as the *original* Hankel singular values.

5.1. Rail Model

Figures 1 and 2 report on the rail model. We nicely see the approximation of the Hankel singular values from below. Especially we see that after only 20 dual ADI steps we can no longer distinguish the original and approximate values in the eyeball norm.

To show how fast the singular values converge we plotted the relative change of the leading values up to iteration 61. Here the process stopped since the largest relative change already dropped below machine precision. Another effect that is often observed can also be seen. The convergence is not monotone. We clearly see that for some values

Figure 2: relative change in Hankel singular values after different numbers of dual ADI iterations

the relative change in step 20 is even larger than in step 15. This is due to the different influences of the shift parameters in the ADI process. A good shift may give a better contribution to the error reduction than a bad one. Then again if the error is more drastically reduced in some steps, the relative change may be larger than in steps where the contribution of the shift is fairly poor.

5.2. BIPS07_1693

For the BIPS model we investigate some more things. In Figure 3 we repeat the test from the previous section. Obviously the convergence here is a lot slower. This is due to several facts. First of all we already mentioned the very bad conditioning of the A_{22} block, but additionally this system is very close to instability making it especially tough for any Lyapunov equation solver, since the Lyapunov operator is close to singular itself.

A second observation is the bad reliability of residuals in combination with balanced truncation. Figure 4 clearly shows that residuals would have told us to stop the iteration after roughly 30 steps due to stagnation. This compares to the circles in Figure 3, where we see that not even one of the singular values is matched. Note that the relative model reduction error is smaller than 10% for a wide range of frequencies anyway.

Finally we report on the speedups possible by our approach in the Table 1.

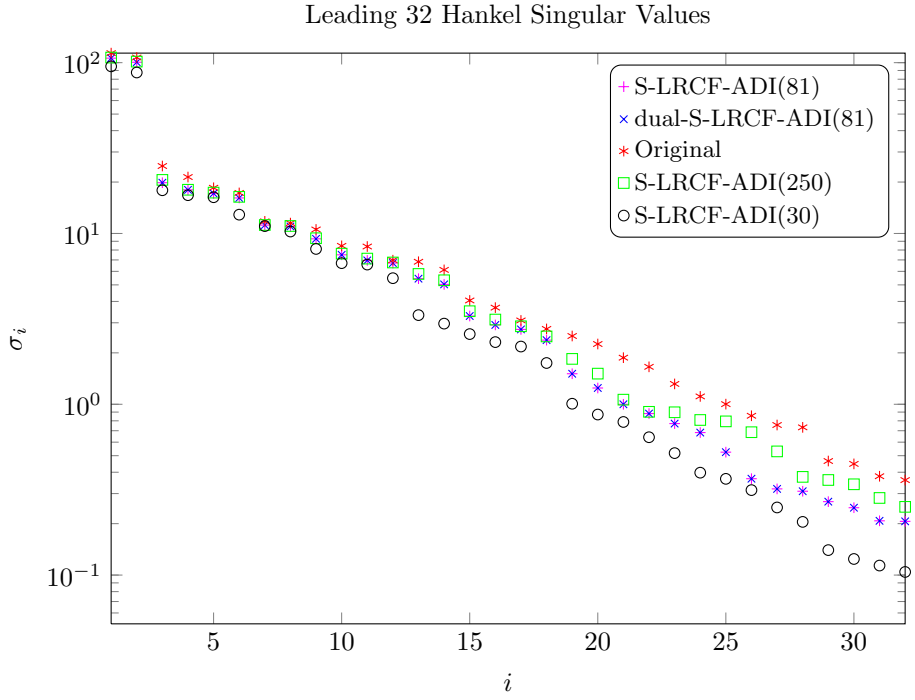

Figure 3: Hankel singular values for several low rank Gramian approximations

maxiter	50	75	100	125
t_{dual}	2.87	4.60	7.15	10.13
$t_{\text{S-LRCF-ADI}}$	24.12	33.14	43.71	54.17
speedup	8.42	7.21	6.11	5.35
change HSV	1.150 e-01	6.740 e-02	6.229 e-02	5.884 e-02
final res. B	5.714 e-09	1.344 e-09	6.134 e-10	1.250 e-10
final res. C	2.004 e-03	2.359 e-03	2.368 e-03	2.341 e-03

	150	175	200	225	250
	14.88	19.67	32.93	38.57	52.98
	66.27	85.23	90.51	102.5	112.7
	4.45	4.33	2.75	2.65	2.13
	5.839 e-02	5.820 e-02	5.815 e-02	5.812 e-02	5.811 e-02
	1.440 e-10	9.196 e-11	9.461 e-10	9.259 e-11	2.049 e-11
	2.334 e-03	2.332 e-03	2.331 e-03	2.331 e-03	2.331 e-03

Table 1: Execution time (in s) comparison for the dual and the two single iterations.

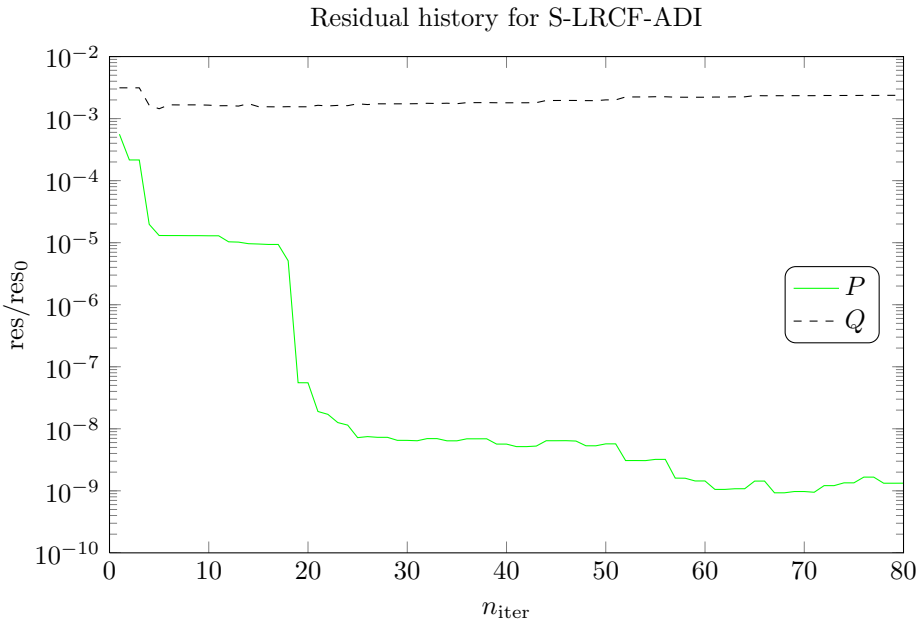

Figure 4: Residual history over the first 80 iterations.

Again we observe that residuals do not tell us the truth about the actual convergence of the subsystems we want to match. The change of Hankel singular values stays at an almost constant level implying that we are still converging whereas the residuals stagnate, telling that we do not add any essential information regarding the Lyapunov solution. The bad news here is that the increasing cost of the SVD computations kills the performance gain when the iteration has to run for many steps and thus the matrix in the SVD gets large.

6. Conclusions

We have seen that tackling the two dual Lyapunov equations simultaneously can give a considerable speedup in the computation time. This is especially due to the ability to use problem tailored stopping criteria that may be much more efficient to compute than the Lyapunov residuals. Also we have seen that this way we can highly increase the reliability of when to stop the iteration. The main recommendation resulting from this contribution is therefore to use tailored stopping criteria in any application where it is possible when working with low rank Lyapunov solution factors.

References

[1] R. Aldhaheri, *Model order reduction via real Schur-form decomposition*, Internat. J. Control, 53 (1991), pp. 709–716.

[2] A. C. Antoulas, *Approximation of Large-Scale Dynamical Systems*, SIAM Publications, Philadelphia, PA, 2005.

[3] P. Benner, M. Köhler, and J. Saak, *Sparse-dense sylvester equations in $\mathcal{H}_2$-model order reduction.*, Tech. Rep. MPIMD/11-11, Max Planck Institute Magdeburg Preprints, December 2011. submitted to Journal of computational and Applied Mathematics.

[4] P. Benner, J.-R. Li, and T. Penzl, *Numerical solution of large Lyapunov equations, Riccati equations, and linear-quadratic control problems*, Numer. Lin. Alg. Appl., 15 (2008), pp. 755–777.

[5] K. Fernando and H. Nicholson, *On a fundamental property of the cross-Gramian matrix*, IEEE Trans. Circuits Syst., CAS-31 (1984), pp. 504–505.

[6] ——, *On the structure of balanced and other principal representations of linear systems*, IEEE Trans. Automat. Control, AC-28 (1984), pp. 228–231.

[7] F. Freitas, J. Rommes, and N. Martins, *Gramian-based reduction method applied to large sparse power system descriptor models*, IEEE Trans. Power Systems, 23 (2008), pp. 1258–1270.

[8] G. Golub and C. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, third ed., 1996.

[9] A. J. Laub, M. T. Heath, C. C. Paige, and R. C. Ward, *Computation of system balancing transformations and other applications of simultaneous diagonalization algorithms.*, IEEE Trans. Autom. Control, 32 (1987), pp. 115–122.

[10] J.-R. Li, *Model Reduction of Large Linear Systems via Low Rank System Gramians*, PhD thesis, Massachusettes Institute of Technology, September 2000.

[11] B. C. Moore, *Principal component analysis in linear systems: controllability, observability, and model reduction*, IEEE Trans. Automat. Control, AC-26 (1981), pp. 17–32.

[12] J. Saak, *Efficient Numerical Solution of Large Scale Algebraic Matrix Equations in PDE Control and Model Order Reduction*, PhD thesis, TU Chemnitz, July 2009. Available from `http://nbn-resolving.de/urn:nbn:de:bsz:ch1-200901642`.

[13] M. Tombs and I. Postlethwaite, *Truncated balanced realization of a stable non-minimal state-space system*, Internat. J. Control, 46 (1987), pp. 1319–1330.

[14] Y. Zhou and D. C. Sorensen, *Approximate implicit subspace iteration with alternating directions for LTI system model reduction*, Numer. Lin. Alg. Appl., 15 (2008), pp. 873–886. DOI: 10.1002/nla.602.

A. A Sample MATLAB Implementation

The following code listing represents the generalized case. The implementation for the standard case can be achieved by setting $M = I$. An efficient implementation should however replace M in the code to avoid unnecessary multiplications by the identity matrix saving $(kg + kh + zg + zh) * n$ flops in each iteration. Here kg, kh are the number of columns and rows in G and H respectively and zg, zh are the corresponding dimensions in the computed low rank factors ZG and ZH.

```
function [ZG,ZH,sd,niter]=lrcf_g_adi_dual_bt( ...
             F,M,G,H,p,maxiter,rtol,k,issym, verb)
% function [ZG,ZH,resg,resh,niter] =
%              lrcf_adi_r_dual(F,G,p,maxiter,rtol,k
%              ,issym)
%
% Generate low rank matrices ZG, ZH such that
% P=ZG*ZG.', Q=ZH*ZH.' solve the Lyapunov equations:
%
%  F*P*M' + M*P*F' = -G*G'                    (1)
% F'*Q*M  + M*Q*F  = -H'*H                    (2)
%
% The function implements the low rank Cholesky
% factor ADI method as proposed by
% Benner/Li/Penzl[1] (Algorithm 3) extended to the
% case of generalized systems taking M (invertible)
% instead of M=I.
%
% Inputs:
% F,M,G,H    The matrices F,G,H in (1), (2).
%  p         a vector of shift parameters
% maxiter    maximum iteration number.
% rtol       tolerance for the residual norm based
%            stopping criterion
%  k         desired reduced model order
% issym      symmetrie flag for the operator F
%
% Outputs:
% ZG,ZH      The solution factors of P and Q in
%            (1) or (2) respectively.
% sd         the vector of relativ changes in HSVs.
% niter      the number of iteration steps taken

% input parameters not checked!

n=size(F,1);

sigma=ones(k,1);
sigmadiff=1;
```

```
sd=zeros(2,maxiter);

kg=size(G,2);
kh=size(H,1);
l=length(p);

V1G=zeros(n,kg);
V1H=zeros(n,kh);

pc=p(1);

if issym
  V1=sqrt(-2*real(pc))*((F+pc*M)\[G H']);
  ZG=V1(:,1:kg);
  ZH=V1(:,kg+1:kg+kh);
else
  [LS,US,pa,qa]=lu(F+pc*M,'vector');
  y=LS\G(pa,:);
  z(qa,:)=US\(y);
  V1G=sqrt(-2*real(pc))*z;
  y=US.'\(H(:,qa)');
  z=[];
  z(pa,:)=LS.'\(y);
  V1H=sqrt(-2*real(pc))*z;
  ZG=V1G;
  ZH=V1H;
end


for i=2:maxiter
  ip=mod(i+l-1,l)+1; pp=pc;  pc=p(ip);
  if issym
    V1=sqrt(real(pc)/real(pp))*(V1 ...
       -(pc+conj(pp))*((F+pc*M)\(M*V1)));
    ZG=[ZG V1(:,1:kg)];
    ZH=[ZH V1(:,kg+1:kg+kh)];
  else
    [LS,US,pa,qa]=lu(F+pc*M,'vector');

    rhsG=M*V1G;
    y =  LS\(rhsG(pa,:));
    z=[];
    z(qa,:) = US\(y);
    V1G=sqrt(real(pc)/real(pp))*(V1G ...
        -(pc+conj(pp))*z);
    ZG=[ZG V1G];

    rhsH=M'*V1H;
    y=US.'\rhsH(qa,:);
    z=[];
    z(pa,:)= LS.'\(y);
    V1H=sqrt(real(pc)/real(pp))*(V1H ...
        -(pc+conj(pp))*z);
    ZH=[ZH V1H];
  end
```

```
  if size(ZH,2)≥k && size(ZG,2)≥k
    sigmaold = sigma;
    sigma = svd(ZH'*(M*ZG));
    sigma = sigma(1:k);
    sigmadiff = abs(sigma—sigmaold)/sigma(1);
    sd(1,i)=max(sigmadiff);
    sd(2,i)=min(sigmadiff);
  end

  if max(sigmadiff)<rtol, break; end

end

niter=i;
sd=sd(:,1:niter);
```