Factorized solution of Lyapunov equations based on hierarchical matrix arithmetic*

Ulrike Baur† and Peter Benner‡

October 6, 2004

Abstract

We investigate the numerical solution of large-scale Lyapunov equations with the sign function method. Replacing the usual matrix inversion, addition, and multiplication by formatted arithmetic for hierarchical matrices, we obtain an implementation that has linear-polylogarithmic complexity and memory requirements. The method is well suited for Lyapunov operators arising from FEM and BEM approximations to elliptic differential operators. With the sign function method it is possible to obtain a low-rank approximation to a full-rank factor of the solution directly. The task of computing such a factored solution arises, e.g., in model reduction based on balanced truncation. The basis of our method is a partitioned Newton iteration for computing the sign function of a suitable matrix, where one part of the iteration uses formatted arithmetic while the other part directly yields approximations to the full-rank factor of the solution. We discuss some variations of our method and its application to generalized Lyapunov equations. Numerical experiments show that the method can be applied to problems of order up to $\mathcal{O}(10^5)$ on desktop computers.

1 Introduction

This paper is concerned with the numerical solution of *Lyapunov equations* of the form

$$AX + XA^T + BB^T = 0 \tag{1}$$

with the coefficient matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, and the solution matrix $X \in \mathbb{R}^{n\times n}$.

Many of the applications of Lyapunov equations arise from analysis and control design problems for linear time-invariant systems of the form

$$\dot{x}(t) = Ax(t) + Bu(t), \quad x(0) = x_0, \tag{2}$$

*Supported by the DFG Research Center "Mathematics for key technologies" (Matheon) in Berlin.

†e-mail: baur@math.tu-berlin.de, Phone: +49 (0)30 314 79177, Fax: +49 (0)30 314 79706, Technische Universität Berlin, Institut für Mathematik, Straße des 17. Juni 136, 10623 Berlin, Germany

‡e-mail: benner@mathematik.tu-chemnitz.de, Phone: +49 (0)371 531 8367, Fax: +49 (0)371 531 2657, Technische Universität Chemnitz, Fakultät für Mathematik, Reichenhainer Straße 41, 09107 Chemnitz, Germany

where $u(t) \in \mathbb{R}^m$ is the vector of input variables, and $x(t) \in \mathbb{R}^n$ denotes the vector of state variables, see, e.g., [1, 15, 16, 39, 46]. Usually, the definition of a linear dynamical system includes an additional output equation. This equation leads to a dual Lyapunov equation, which can be addressed by analog Lyapunov solvers and is therefore neglected for the purpose of this paper.

Often, in practice, e.g., in the control of partial differential equations (PDEs), the system matrix A comes from the discretization of some partial differential operator. In this case, n, the dimension of the state space, is typically large (often $n \geq \mathcal{O}(10^4)$) and the system matrices are sparse. On the other hand, boundary element discretizations of integral equations lead to large-scale *dense* systems that often have a data-sparse representation [25]. Usually, the number of inputs in practical applications is small compared to the number of states, so that it is reasonable to assume $m \ll n$ for the rest of this paper. Moreover, when A represents the approximation of an elliptic differential operator, it is often a (*Hurwitz) stable* matrix, that is, all its eigenvalues, denoted by $\Lambda(A)$, are contained in the open left half plane $\mathbb{C}^-$. For instance, when (2) comes from the spatial semi-discretization of the instationary linear heat equation, A is

- (a scalar multiple of) the discrete Laplacian if finite differences are used;
- (formally) equal to $-M^{-1}K$, where M and K are mass and stiffness matrices, if a finite element discretization is used.

Apart from situations leading to singularity of A, in both cases, A is (similar to) a negative definite matrix and hence has real negative eigenvalues. In the following, we will therefore always assume stable A matrices.

The stability assumption on A together with the positive semi-definiteness of the "right-hand side term" BB^T implies that the Lyapunov equation (1) has a unique, symmetric nonnegative definite solution X [35]. Hence, it can be factored as $X = YY^T$. Possibilities for Y are

- the *Cholesky factor* of X, i.e., $Y \in \mathbb{R}^{n\times n}$ is a square lower triangular matrix,
- a *full-rank factor* of X, i.e., $Y \in \mathbb{R}^{n\times \operatorname{rank}(X)}$ is a rectangular matrix.

The latter option is of particular interest for large-scale computations if X has low rank, $n_X := \operatorname{rank}(X) \ll n$, as (1) represents a linear system of equations with $n(n+1)/2$ unknowns (exploiting symmetry). In such a situation, the memory requirements for storing X can be considerably reduced by working with Y instead of X. Interpreting "$\operatorname{rank}(X)$" as numerical rank [18] (or ε-rank), it is often the case that this numerical rank is very low even though theoretically, X may be nonsingular. In that case, using a spectral (or singular value) decomposition of X, it is easy to see that Y can be approximated by a "tall" matrix $\hat{Y} \in \mathbb{R}^{n\times n_Y}$, $n_Y \ll n$, so that

$$\frac{\|X - \hat{Y}\hat{Y}^T\|_2}{\|X\|_2} \leq \varepsilon$$

with the tolerance threshold ε determining the numerical rank. In many large-scale applications it can be observed that the eigenvalues of X decay rapidly, so that a low-rank approximation in the form described above exists; see [2, 20, 43].

This observation has led to various approaches for solving Lyapunov equations by methods based on an approximate low-rank factorization of the solution [8, 37, 43] and is also the basis of several multigrid methods for solving (1) [22, 40, 45].

In [10, 36, 42], low-rank (approximate) factors are used for model reduction based on balanced truncation. Model reduction aims at approximating a large-scale system of the form (2) by a system of much smaller dimension $r \ll n$. Balanced truncation [38] is one of the most commonly used model reduction methods for linear time-invariant systems [1, 39], and requires the solution of the two dual Lyapunov equations corresponding to the linear system (1) as a first computational step. The common characteristic of the methods developed in [10, 36, 42] is that using the low-rank solution factors, all further computational steps of balanced truncation only require only $\mathcal{O}(n_Y^2 n)$ floating-point operations (flops) rather than the $\mathcal{O}(n^3)$ flops needed in standard implementations as contained, e.g., in SLICOT [7, 49]. Thus, for an efficient balanced truncation implementation for large-scale systems, it is crucial to have Lyapunov solvers that are able to compute $\hat{Y}$ directly without ever forming X.

The standard direct method for solving Lyapunov equations is the Bartels-Stewart method [5] and the direct computation of Cholesky factors of the solution via this approach is suggested by Hammarling in [28]. But since this method requires $\mathcal{O}(n^3)$ flops and $\mathcal{O}(n^2)$ memory, it is only practicable for problems of relatively small size. Apart from direct methods, there are several iterative methods, for example the Smith method [47], the alternating direction implicit iteration (ADI) method [50], and the Smith(l) method [42]. These methods can be modified to compute $\hat{Y}$, see [37, 24, 43] and are therefore viable approaches to be used in large-scale applications, see also [3]. There are also several approaches to solve large-scale Lyapunov equations using Krylov subspace methods [31, 32, 33, 34], but in general they are inferior to ADI and Smith-type methods, see [41].

In this paper, we will propose a new method based on the *sign function method*, published first in 1971 by Roberts [44], incorporating the idea of computing low-rank factors of the solution as suggested in [8]. Despite the low memory requirements for $\hat{Y}$, this method still needs $\mathcal{O}(n^2)$ storage and is therefore of limited use for really large-scale problems, though it parallelizes well [6]. In [23], Grasedyck, Hackbusch and Khoromskij combine the hierarchical matrix ($\mathcal{H}$-matrix) format with the sign function method for solving algebraic Riccati equations (AREs) to avoid this limitation. The $\mathcal{H}$-matrix format is described, e.g., in [19, 21, 26, 27]; it allows data-sparse approximation for a wide, practically relevant class of matrices, which, e.g., arise from boundary element or finite element methods. The matrices during the sign function iteration and the solution itself are approximated in $\mathcal{H}$-matrix format. Using an appropriate, formatted arithmetic leads to a variant of Roberts' method that has linear-polylogarithmic complexity. As the Lyapunov equation is a special (simplified) version of an ARE, in principal this method could be applied directly to (1), but it does not provide the factor $\hat{Y}$ needed, e.g., in the balanced truncation implementations mentioned above.

To obtain an (approximate) full-rank factor of X we consider the sign function iteration in partitioned form, as proposed in [8]. During the sign function iteration we have to take care of the following fact. If we consider matrices resulting from finite element discretizations of elliptic partial differential oper-

ators, we are dealing with matrices in sparse form. Applying the sign function iteration to these matrices, we have to compute the inverse of A, which destroys the assumed sparsity of A. To avoid this, the matrix A and its inverse are approximated in the $\mathcal{H}$-matrix format and the corresponding approximate arithmetic is used for driving the iteration.

This paper is organized as follows: in Section 2, we describe the sign function iteration for the solution of Lyapunov equations. Some basic facts of the $\mathcal{H}$-matrix format and the corresponding formatted arithmetic are given in Section 3. Two variants of a $\mathcal{H}$-sign function method for Lyapunov equations and numerical experiments demonstrating the performance of the new algorithm are described in 4. In Section 5, we extend the derived results to the generalized Lyapunov equation, which is of interest in control theory, when the control problem is governed, e.g., by second-order (instead of first-order) ordinary differential equations [8].

2 Lyapunov equation and sign function iteration

One of the numerical methods to address the Lyapunov equation (1) is based on the sign function method [44].

To describe this method, consider a matrix $Z \in \mathbb{R}^{n\times n}$ with no eigenvalues on the imaginary axis. By the real version of the Jordan canonical form there exists a nonsingular matrix $S \in \mathbb{R}^{n\times n}$ s.t.

$$Z = S^{-1} \begin{bmatrix} J_l^+ & 0 \\ 0 & J_{n-l}^- \end{bmatrix} S,$$

where $\Lambda\left(J_l^+\right) \subset \mathbb{C}^+$, $\Lambda\left(J_{n-l}^-\right) \subset \mathbb{C}^-$. Then the *matrix sign function* for Z is defined by

$$\operatorname{sign}(Z) := S^{-1} \begin{bmatrix} I_l & 0 \\ 0 & -I_{n-l} \end{bmatrix} S.$$

To compute the matrix sign function, we use the Newton iteration applied to $(\operatorname{sign}(Z))^2 = I_n$:

$$Z_0 \leftarrow Z, \qquad Z_{k+1} \leftarrow \frac{1}{2}(Z_k + Z_k^{-1}).$$

This so called sign function iteration converges globally quadratically to the sign of Z and is well-behaved in finite-precision arithmetic [13].

In order to solve the Lyapunov equation (1), we apply this iteration to the particular matrix:

$$Z = \begin{bmatrix} A & BB^T \\ 0 & -A^T \end{bmatrix} \tag{3}$$

and obtain the following iteration scheme:

$$\begin{aligned} Z_0 &\leftarrow Z, \\ Z_{k+1} &\leftarrow \frac{1}{2}(Z_k + Z_k^{-1}) \\ &= \begin{bmatrix} \frac{1}{2}(A_k + A_k^{-1}) & \frac{1}{2}(B_k B_k^T + A_k^{-1} B_k B_k^T A_k^{-T}) \\ 0 & -\frac{1}{2}(A_k + A_k^{-1})^T \end{bmatrix}. \end{aligned}$$

The solution X of (1) can then be derived by

$$\operatorname{sign}(Z) = \lim_{k\to\infty} Z_k = \begin{bmatrix} -I_n & 2X \\ 0 & I_n \end{bmatrix}$$

as described in [44].

To accelerate the initial convergence, some of the iterates can be scaled in the following way:

$$Z_{k+1} \leftarrow \frac{1}{2}(c_k Z_k + \frac{1}{c_k} Z_k^{-1}),$$

where $c_k > 0$ are suitable chosen parameters. Several choices for such parameters can be found in, e.g., [4, 12].

In certain applications, such as model reduction by balanced truncation, we are more interested in computing a full-rank factor Y, s.t. $X = YY^T$. To obtain the factorized solution, we partition the iteration into two parts:

$$\begin{aligned} A_0 &\leftarrow A, \quad B_0 \leftarrow B, \\ A_{k+1} &\leftarrow \frac{1}{2}(A_k + A_k^{-1}), \\ B_{k+1} &\leftarrow \frac{1}{\sqrt{2}} \begin{bmatrix} B_k & A_k^{-1} B_k \end{bmatrix}, \quad k = 1, 2, \ldots. \end{aligned}$$

see [8] for details. The matrix $Y = \frac{1}{\sqrt{2}} \lim_{k\to\infty} B_k$ is a factor of the solution

$$X = YY^T = \frac{1}{2} \lim_{k\to\infty} B_k B_k^T.$$

Since the size of the matrix B_{k+1} in (4) is doubled in each iteration step, it is proposed in [8] to apply a rank-revealing QR factorization (RRQR) [18] to B_{k+1}^T in order to limit the exponentially growing number of columns:

$$B_{k+1}^T = Q_{k+1} R_{k+1} P_{k+1} = Q_{k+1} \begin{bmatrix} R_{k+1}^{11} & R_{k+1}^{12} \\ 0 & R_{k+1}^{22} \end{bmatrix} P_{k+1}. \tag{4}$$

Here P_{k+1} is a permutation matrix, Q_{k+1} is orthogonal, R_{k+1}^{11} is a $\mathbb{R}^{r_{k+1}\times r_{k+1}}$ matrix (r_{k+1} denotes the numerical rank of B_{k+1}^T), while R_{k+1}^{22} is of small norm. Only the entries in the upper triangular part of R_{k+1} have to be stored for obtaining an approximate solution $\hat{Y} = \frac{1}{\sqrt{2}} \lim_{k\to\infty} \hat{B}_k$, with

$$\hat{B}_{k+1}^T := \begin{bmatrix} R_{k+1}^{11} & R_{k+1}^{12} \end{bmatrix} P_{k+1}.$$

3 $\mathcal{H}$-matrix arithmetic

In [23], the sign function method for solving the more general algebraic Riccati equation was combined with a data-sparse matrix representation and a corresponding approximate arithmetic. As our approach also makes use of this $\mathcal{H}$-matrix format, we will introduce some of its basic facts in the following.

The $\mathcal{H}$-matrix format is a data-sparse representation for a special class of matrices, which often arise in applications. Matrices that belong to this class result, for instance, from the discretization or linearization of partial differential

or integral equations. Exploiting the special structure of these matrices in computational methods yields decreased time and memory requirements. A detailed description of the $\mathcal{H}$-matrix format can be found, e.g. in [19, 21, 26, 27].

The basic idea of the $\mathcal{H}$-matrix format is to partition a given matrix recursively into submatrices that admit low-rank approximations. To determine such a partitioning, we consider a product index set $I \times I$, $I = \{1, \ldots, n\}$. This product index set is hierarchically partitioned into blocks $r \times s$, which form a so called $\mathcal{H}$-tree $T_{I\times I}$. Each leaf of $T_{I\times I}$ represents a low-rank approximation of the corresponding submatrix. A matrix M is said to be approximable in $\mathcal{H}$-matrix format ($M \in \mathcal{M}_{\mathcal{H},k}(T_{I\times I})$), if the rank of M restricted to a leaf can be bounded by k. The storage requirements for a matrix $M \in \mathcal{M}_{\mathcal{H},k}(T_{I\times I})$ are

$$\mathcal{N}_{\mathcal{M}_{\mathcal{H},k}St} = \mathcal{O}(n \log(n) k)$$

instead of $\mathcal{O}(n^2)$ for the original matrix.

Note that it is also possible to choose the rank adaptively for each matrix block instead of using a fixed rank k. Depending on a given approximation error ϵ, the approximate matrix operations are exact up to ϵ in each block.

The approximate arithmetic is a means to close the set of matrices in $M \in \mathcal{M}_{\mathcal{H},k}(T_{I\times I})$ under addition, multiplication and inversion. The operations consist of the exact arithmetic combined with some projection onto $\mathcal{M}_{\mathcal{H},k}(T_{I\times I})$. This truncation operator, denoted by $\mathcal{T}_k$, can be achieved by truncated singular value decompositions and results in the best Frobenius norm approximation on $\mathcal{M}_{\mathcal{H},k}(T_{I\times I})$, see, e.g., [21] for more details. For two matrices $A, B \in \mathcal{M}_{\mathcal{H},k}(T_{I\times I})$ and a vector $v \in \mathbb{R}^n$ we obtain the following formatted arithmetic operations, which all have linear-polylogarithmic complexity:

$$\begin{array}{lcll} v \mapsto Av: & & & \mathcal{O}(n\log(n)k), \\ A \oplus B & = & \mathcal{T}_k(A+B): & \mathcal{O}(n\log(n)k^2), \\ A \odot B & = & \mathcal{T}_k(AB): & \mathcal{O}(n\log^2(n)k^2), \\ \mathrm{Inv}_{\mathcal{H}}(A) & = & \mathcal{T}_k(\tilde{A}^{-1}): & \mathcal{O}(n\log^2(n)k^2). \end{array}$$

Here, $\tilde{A}^{-1}$ denotes the approximate inverse of A which is obtained by performing block Gauss elimination on A with formatted addition and multiplication.

We will use the $\mathcal{H}$-matrix structure to compute the solution factor of the Lyapunov equation, which reduces the complexity and the storage requirements of the sign function iteration.

4 $\mathcal{H}$-matrix arithmetic based sign function iteration

4.1 Algorithms

We consider the sign function iteration in the partitioned form (4), in contrast to [23], to compute a full-rank factor Y of the solution X of (1). In one part of the iteration the hierarchical matrix arithmetic is integrated to reduce memory requirements and computational costs (compare with Section 3). In this part, even if the system matrix A is sparse, a larger amount of memory is required by the fill-in during the matrix inversion. The other part of the iteration is stored in the usual "full" format and uses arithmetic operations from standard

linear algebra packages like LAPACK and BLAS. This part converges to $Y = \frac{1}{\sqrt{2}} \lim_{k\to\infty} B_k$, which is an approximate full-rank factor of X. The increasing number of columns of B_{k+1} again is limited by applying the rank-revealing QR factorization as in Section 2. Since $\lim_{k\to\infty} A_k = -I_n$, as it was seen in Section 2, it is advised to choose

$$\|A_k + I_n\| \leq \text{tol}$$

as stopping criterion for the iteration, which is easy to check. With two additional iteration steps and an appropriate choice of norm and relaxed tolerance, the required accuracy is reached in general due to the quadratic convergence, see [8] for details. We introduce scaling to accelerate the initial convergence. Due

Algorithm 1 Calculate full-rank factor Y of X for $AX + XA^T + BB^T = 0$

INPUT: $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, tol, ε
OUTPUT: Approximation to a full-rank factor of the solution X.

```
A_0 ← (A)_H
B_0 ← B
k = 0
while ‖A_k + I_n‖ > tol do
    A_{k+1} ← (1/2)(A_k ⊕ Inv_H(A_k))
    B_{k+1} ← (1/√2) [ B_k   Inv_H(A_k)B_k ]
    Compress columns of B_{k+1} (see (4)) using a RRQR with threshold ε
    k = k + 1
end while
Y ← (1/√2) B_{k+1}
```

to error amplification during the sign function iteration with formatted arithmetic, scaling is used only in the first iteration step as in [19]. In the partitioned iteration scheme 1 scaling is integrated in the following way:

$$\begin{aligned} A_0 &\leftarrow (A)_{\mathcal{H}}, \quad B_0 \leftarrow B, \\ A_1 &\leftarrow \frac{1}{2}(c_0 A_0 \oplus \frac{1}{c_0}\text{Inv}_{\mathcal{H}}(A_0)), \\ B_1 &\leftarrow \frac{1}{\sqrt{2}} c_0 \begin{bmatrix} B_0 & \text{Inv}_{\mathcal{H}}(A_0)B_0 \end{bmatrix} \end{aligned}$$

with scaling parameter c_0 as proposed in [19].

An alternative algorithm replaces the formatted inversion by computing a LU decomposition of the matrix A_k. The lower and upper parts are stored in $\mathcal{H}$-format and with an $\mathcal{H}$-based forward substitution we obtain an approximate inverse of A_k.

Algorithm 2 Calculate full-rank factor Y of X for $AX + XA^T + BB^T = 0$

INPUT: $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, tol, ε
OUTPUT: Approximation to a full-rank factor of the solution X.

```
A_0 ← (A)_H
B_0 ← B
k = 0
while ‖A_k + I_n‖ > tol do
  [L, U] ← LU_H(A_k)
  Solve LW = (I_n)_H by H-forward substitution
  Solve UV = W by H-back substitution
  A_{k+1} ← 1/2 (A_k ⊕ V)
  B_{k+1} ← 1/√2 [ B_k  VB_k ]
  Compress columns of B_{k+1} (see (4)) using a RRQR with threshold ε
  k = k + 1
end while
Y ← 1/√2 B_{k+1}
```

4.2 Numerical experiments

We consider the two-dimensional heat equation in an unit square with constant heat source in some subdomain Ω_u as described in [23]:

$$\begin{aligned}\frac{\partial \mathbf{x}}{\partial t}(t,\xi) &= \frac{\lambda}{c\cdot\rho}\,\Delta\mathbf{x}(t,\xi) + b(\xi)u(t), \qquad \xi \in (0,1)^2, t \in (0,\infty)\\ b(\xi) &= \begin{cases} 1 & \xi \in \Omega_u \\ 0 & \text{otherwise}\end{cases}.\end{aligned}$$

We impose homogeneous Dirichlet boundary conditions

$$\mathbf{x}(t,\xi) = 0 \quad \xi \in [0,1]^2 \backslash (0,1)^2$$

and discretize with linear finite elements and n inner grid points. In the weak form of the partial differential equation we use a classical Galerkin approach with bilinear finite ansatz functions φ_i: $\mathbf{x}(t,\xi) = \sum_{i=1}^{n} x_i(t)\varphi_i(\xi)$. For the n unknowns x_i we obtain a system of linear differential equations

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

with the matrices

$$\begin{aligned}E_{ij} &= \int_{(0,1)^2} \varphi_i(\xi)\varphi_j(\xi)\,d\xi\\ A_{ij} &= -\int_{(0,1)^2} \lambda\,\nabla\varphi_i(\xi)\cdot\nabla\varphi_j(\xi)\,d\xi\\ B_{i1} &= \int_{(0,1)^2} b(\xi)\varphi_i(\xi)\,d\xi, \quad i,j = 1,\ldots,n.\end{aligned}$$

To obtain a system in the standard form (2) we have to invert the mass matrix E, what is done with the formatted inversion, and apply algorithm 1 or 2 to the matrices

$$A = \mathrm{Inv}_{\mathcal{H}}(E)A, \quad B = \mathrm{Inv}_{\mathcal{H}}(E)B.$$

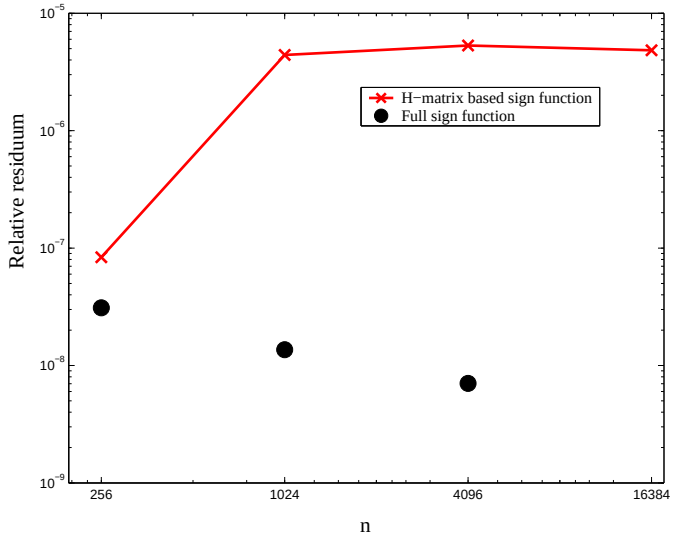

Figure 1: Relative residual in logarithmic scale for the $\mathcal{H}$-matrix based sign function and the usual full sign function

In the first iteration step, we use scaling as proposed in [29, 19] with

$$c_0 = \sqrt{\frac{\|\mathrm{Inv}_{\mathcal{H}}(A_0)\|_2}{\|A_0\|_2}}.$$

The employed stopping criterion for the Newton iteration is:

$$\|A_k + I\|_2 \leq \mathrm{tol}, \quad \mathrm{tol} = 10^{-4}.$$

We choose $\varepsilon = 10^{-4}$ as threshold for the numerical rank decision in the rank-revealing QR factorization.

For the $\mathcal{H}$-matrix approximation we use HLib 1.2 by Börm, Grasedyck, Hackbusch [11]. We use the adaptive rank choice (see [19]) instead of a given rank k. The truncation operator is then changed in the following way:

$$\mathcal{T}_\epsilon(A) = \mathrm{argmin}\{\mathrm{rank}(R) | \frac{\|R - M\|_2}{\|M\|_2} \leq \epsilon\},$$

where the parameter ϵ is given by $\epsilon = 10^{-4}$ and determines the desired accuracy in each matrix block.

These results are obtained by use of Algorithm 1. The sign function without $\mathcal{H}$-matrix implementation can be used only up to a problem size of $n = 4096$ due to memory requirements, larger problems can only be solved with the $\mathcal{H}$-matrix based sign function. In Figure 4.2 we observe, that the relative residual, which could be considered as the backward error for the solution of the Lyapunov equations [30], seems to be bounded above for increasing problem size. The storage requirements as well as the computational time for Algorithm 1 exhibit

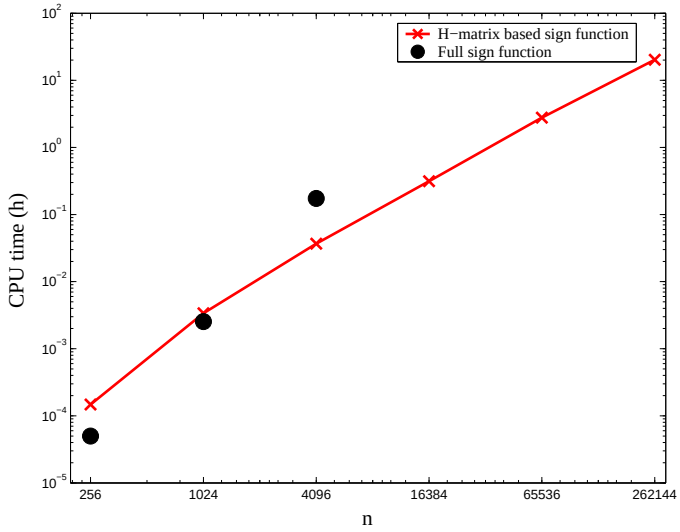

Figure 2: CPU time in logarithmic scale for the $\mathcal{H}$-matrix based sign function and the usual full sign function

an almost linear complexity as can be seen in Figure 4.2 and in Table 1. We want to point out, that the largest Lyapunov equations solved, one with $n = 262,144$, is equivalent to a linear system of equations with about 34 billion unknowns. For this problem size we get an approximate full-rank factor $Y \in \mathbb{R}^{n\times 21}$ and therefore need 5 MB memory to store the solution instead of 64 GB for the explicit solution X.

	n	r	time[sec]	memory (MB)	rel. res.	rel. error
full	256	11	0.18	0.5	3.096e-08	
$\mathcal{H}$		11	0.53	0.48	8.362e-08	6.424e-07
full	1024	13	9.12	8.00	1.361e-08	
$\mathcal{H}$		13	12.17	4.21	4.407e-06	6.302e-05
full	4096	14	624.21	128.00	7.035e-09	
$\mathcal{H}$		14	132.19	29.47	5.310e-06	1.612e-04
$\mathcal{H}$	16384	15	1129.94	192.86	4.831e-06	-
$\mathcal{H}$	65536	17	10002.09	1019.65	-	-
$\mathcal{H}$	262144	21	72910.44	4431.62	-	-

Table 1: The table presents the accuracy and the rank r of the computed solution factor for different problem sizes. Also the different memory requirements for storing A in $\mathcal{H}$-format or in full-format can be compared for the last iteration step.

5 Extension to generalized Lyapunov equations

In this section, we show how the derived results can be extended to generalized Lyapunov equations of the form

$$AXE^T + EXA^T + BB^T = 0, \tag{5}$$

where $A, E \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$. Such matrix equations are associated with linear, time-invariant descriptor systems of the form

$$E\dot{x}(t) = Ax(t) + Bu(t), \quad x(0) = x_0,$$

see [14]. Note that (5) reduces to a standard Lyapunov equation if $E = I_n$. Generalized Lyapunov equations with $E \neq I_n$ play an important role in various tasks related to descriptor systems [14], such as minimal realization or balanced truncation model reduction [48].

In the following, we assume that A as well as E are nonsingular and that $A - \lambda E$ is a stable matrix pencil, i.e., all eigenvalues of $A - \lambda E$ are in the open left half plane. Gardiner and Laub [17] proposed an extension of the sign function iteration for the solution of (5). Instead of the single matrix Z in (3), the matrix pencil

$$Z - \lambda Y = \begin{bmatrix} A & BB^T \\ 0 & -A^T \end{bmatrix} - \lambda \begin{bmatrix} E & 0 \\ 0 & E^T \end{bmatrix} \tag{6}$$

is considered. Theoretically, the solution of (5) can be obtained from the $(2,1)$ block of $(Y^{-1}Z)$. Applying the standard sign function iteration directly to $Y^{-1}Z$, however, has the disadvantage that the possibly ill-conditioned matrix E has to be inverted for starting the iteration. An approach which avoids this drawback consists of using the iteration

$$Z_0 \leftarrow Z, \quad Z_{k+1} \leftarrow \frac{1}{2}(Z_k + YZ_k^{-1}Y). \tag{7}$$

It can be easily seen that if $\tilde{Z}_k$ denotes the kth iterate of the standard sign function iteration applied to $Y^{-1}Z$ then $Z_k = Y\tilde{Z}_k$. This implies that the iteration (7) converges under the given assumptions to

$$\lim_{k\to\infty} Z_k = Y \cdot (Y^{-1}Z).$$

In [8] it was shown that the iteration (7) significantly simplifies when applied to a matrix pencil of the form (6):

$$\begin{aligned} Z_0 &\leftarrow Z, \\ Z_{k+1} &\leftarrow \frac{1}{2}(Z_k + YZ_k^{-1}Y) \\ &= \begin{bmatrix} \frac{1}{2}(A_k + EA_k^{-1}E) & \frac{1}{2}(B_kB_k^T + EA_k^{-1}B_kB_k^TA_k^{-T}E^T) \\ 0 & -\frac{1}{2}(A_k^T + E^TA_k^{-T}E^T) \end{bmatrix}. \end{aligned}$$

The solution X of (5) is then obtained by

$$\lim_{k\to\infty} Z_k = \begin{bmatrix} -E & 2EXE^T \\ 0 & E^T \end{bmatrix}.$$

Since we are interested in a full-rank factor Y of the solution X, such that $X = YY^T$, we consider the iteration in factorized form as introduced in [8] for $E \neq I_n$. In this iteration scheme we introduce the hierarchical matrix format and the approximate arithmetic (compare with Section 4.1 for $E = I_n$). As natural stopping criterion ($\lim_{k\to\infty} A_k = -E$) we suggest

$$\|Ak + E\| \leq \text{tol}\|E\|$$

.

Algorithm 3 Calculate full-rank factor Y of X for $AXE^T + EXA^T + BB^T = 0$

INPUT: $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $E \in \mathbb{R}^{n\times n}$, tol, ε
OUTPUT: Approximation to a full-rank factor of the solution X.

```
A_0 ← (A)_H
B_0 ← B
k = 0
while ||A_k + E|| > tol||E|| do
   A_{k+1} ← 1/2 (A_k ⊕ (E)_H ⊙ Inv_H(A_k) ⊙ (E)_H)
   B_{k+1} ← 1/√2 [ B_k   (E)_H Inv_H(A_k) B_k ]
   Compress columns of B_{k+1} (see (4)) using a RRQR with threshold ε
   k = k + 1
end while
Y ← 1/√2 E^{-1} B_{k+1}
```

So we get $Y = \frac{1}{\sqrt{2}} E^{-1} \lim_{k\to\infty} B_k$ as a factor of the approximate solution $X = YY^T$ of (5). In order to accelerate convergence, we choose

$$c = \sqrt{\frac{\|E\text{Inv}_{\mathcal{H}}(A_0)E\|_2}{\|A_0\|_2}}$$

for the first iteration step. This is inspired by the motivation for the scaling in the standard case, numerical results in [9] confirm its ability to accelerate the convergence significantly.

6 Conclusions

In this paper we have developed algorithms for the factorized solution of large Lyapunov equations arising from FEM/BEM discretizations of elliptic partial differential operators. With our $\mathcal{H}$-based sign function approach we can solve significantly larger problems than with the standard dense sign function implementations so that the sign function method becomes competitive with other methods for large-scale problems like ADI and Smith-type methods. This is demonstrated by numerical examples evolving from a 2D heat control problem.

Future work will include to use the developed Lyapunov solvers as building blocks for an implementation of a model reduction method based on balanced truncation for large-scale systems arising from control problems for parabolic PDEs. Variants of our approach for solving Sylvester equations are also under current investigation.

References

[1] A. Antoulas. *Lectures on the Approximation of Large-Scale Dynamical Systems*. SIAM Publications, Philadelphia, PA, to appear.

[2] A. Antoulas, D. Sorensen, and Y. Zhou. On the decay rate of Hankel singular values and related issues. *Sys. Control Lett.*, 46(5):323–342, 2002.

[3] R. Badía, P. Benner, R. Mayo, and E. Quintana-Ortí. Solving large sparse Lyapunov equations on parallel computers. In B. Monien and R. Feldmann, editors, *Euro-Par 2002 – Parallel Processing*, number 2400 in Lecture Notes in Computer Science, pages 687–690. Springer-Verlag, Berlin, Heidelberg, New York, 2002.

[4] Z. Bai and J. Demmel. Design of a parallel nonsymmetric eigenroutine toolbox, Part I. In R. S. et al., editor, *Proceedings of the Sixth SIAM Conference on Parallel Processing for Scientific Computing*, pages 391–398. SIAM, Philadelphia, PA, 1993. *See also:* Tech. Report CSD-92-718, Computer Science Division, University of California, Berkeley, CA 94720.

[5] R. Bartels and G. Stewart. Solution of the matrix equation $AX + XB = C$: Algorithm 432. *Comm. ACM*, 15:820–826, 1972.

[6] P. Benner, J. Claver, and E. Quintana-Ortí. Parallel distributed solvers for large stable generalized Lyapunov equations. *Parallel Processing Letters*, 9(1):147–158, 1999.

[7] P. Benner, V. Mehrmann, V. Sima, S. V. Huffel, and A. Varga. SLICOT - a subroutine library in systems and control theory. In B. Datta, editor, *Applied and Computational Control, Signals, and Circuits*, volume 1, chapter 10, pages 499–539. Birkhäuser, Boston, MA, 1999.

[8] P. Benner and E. Quintana-Ortí. Solving stable generalized Lyapunov equations with the matrix sign function. *Numer. Algorithms*, 20(1):75–100, 1999.

[9] P. Benner, E. Quintana-Ortí, and G. Quintana-Ortí. Solving linear matrix equations via rational iterative schemes. Technical Report SFB393/04-08, Sonderforschungsbereich 393 *Numerische Simulation auf massiv parallelen Rechnern*, TU Chemnitz, 09107 Chemnitz, FRG, 1999. Available from `http://www.tu-chemnitz.de/sfb393/preprints.html`.

[10] P. Benner, E. Quintana-Ortí, and G. Quintana-Ortí. Balanced truncation model reduction of large-scale dense systems on parallel computers. *Math. Comput. Model. Dyn. Syst.*, 6(4):383–405, 2000.

[11] S. Börm, L. Grasedyck, and W. Hackbusch. HLib 1.2, 2004. Available from http://www.numerik.uni-kiel.de/∼lgr/gethmatrix.html.

[12] R. Byers. Solving the algebraic Riccati equation with the matrix sign function. *Linear Algebra Appl.*, 85:267–279, 1987.

[13] R. Byers, C. He, and V. Mehrmann. The matrix sign function method and the computation of invariant subspaces. *SIAM J. Matrix Anal. Appl.*, 18(3):615–632, 1997.

[14] L. Dai. *Singular Control Systems*. Number 118 in Lecture Notes in Control and Information Sciences. Springer-Verlag, Berlin, 1989.

[15] B. Datta. *Numerical Methods for Linear Control Systems Design and Analysis*. Elsevier Press, 2003.

[16] Z. Gajić and M. Qureshi. *Lyapunov Matrix Equation in System Stability and Control*. Math. in Science and Engineering. Academic Press, San Diego, CA, 1995.

[17] J. Gardiner and A. Laub. A generalization of the matrix-sign-function solution for algebraic Riccati equations. *Internat. J. Control*, 44:823–832, 1986.

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

[19] L. Grasedyck. *Theorie und Anwendungen Hierarchischer Matrizen*. Dissertation, University of Kiel, Kiel, Germany, 2001. In German, available at `http://e-diss.uni-kiel.de/diss_454`.

[20] L. Grasedyck. Existence of a low rank or $\mathcal{H}$-matrix approximant to the solution of a Sylvester equation. *Numer. Lin. Alg. Appl.*, 11:371–389, 2004.

[21] L. Grasedyck and W. Hackbusch. Construction and arithmetics of $\mathcal{H}$-matrices. *Computing*, 70(4):295–334, 2003.

[22] L. Grasedyck and W. Hackbusch. A multigrid method to solve large scale Sylvester equations. Preprint 48, Max-Planck Institut für Mathematik in den Naturwissenschaften, Leipzig, Germany, 2004.

[23] L. Grasedyck, W. Hackbusch, and B. Khoromskij. Solution of large scale algebraic matrix Riccati equations by use of hierarchical matrices. *Computing*, 70:121–165, 2003.

[24] S. Gugercin, D. Sorensen, and A. Antoulas. A modified low-rank Smith method for large-scale Lyapunov equations. *Numer. Algorithms*, 32(1):27–55, 2003.

[25] W. Hackbusch. *Integral equations*, volume 120 of *International Series of Numerical Mathematics*. Birkhäuser Verlag, Basel, 1995.

[26] W. Hackbusch. A sparse matrix arithmetic based on $\mathcal{H}$-matrices. I. Introduction to $\mathcal{H}$-matrices. *Computing*, 62(2):89–108, 1999.

[27] W. Hackbusch and B. N. Khoromskij. A sparse $\mathcal{H}$-matrix arithmetic. II. Application to multi-dimensional problems. *Computing*, 64(1):21–47, 2000.

[28] S. Hammarling. Numerical solution of the stable, non-negative definite Lyapunov equation. *IMA J. Numer. Anal.*, 2:303–323, 1982.

[29] N. Higham. Computing the polar decomposition—with applications. *SIAM J. Sci. Statist. Comput.*, 7:1160–1174, 1986.

[30] N. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM Publications, Philadelphia, PA, 1996.

[31] M. Hochbruck and G. Starke. Preconditioned Krylov subspace methods for Lyapunov matrix equations. *SIAM J. Matrix Anal. Appl.*, 16(1):156–171, 1995.

[32] A. Hodel, B. Tenison, and K. Poolla. Numerical solution of the Lyapunov equation by approximate power iteration. *Linear Algebra Appl.*, 236:205–230, 1996.

[33] I. Jaimoukha and E. Kasenally. Krylov subspace methods for solving large Lyapunov equations. *SIAM J. Numer. Anal.*, 31:227–251, 1994.

[34] K. Jbilou and A. Riquet. Projection methods for large Lyapunov matrix equations. *Linear Algebra Appl.*, to appear.

[35] P. Lancaster and M. Tismenetsky. *The Theory of Matrices.* Academic Press, Orlando, 2nd edition, 1985.

[36] J.-R. Li and J. White. Reduction of large circuit models via low rank approximate gramians. *Int. J. Appl. Math. Comp. Sci.*, 11(5):1151–1171, 2001.

[37] J.-R. Li and J. White. Low rank solution of Lyapunov equations. *SIAM J. Matrix Anal. Appl.*, 24(1):260–280, 2002.

[38] B. Moore. Principal component analysis in linear systems: Controllability, observability, and model reduction. *IEEE Trans. Automat. Control*, AC-26:17–32, 1981.

[39] G. Obinata and B. Anderson. *Model Reduction for Control System Design.* Communications and Control Engineering Series. Springer-Verlag, London, UK, 2001.

[40] T. Penzl. A multi-grid method for generalized Lyapunov equations. Technical Report SFB393/97-24, Fakultät für Mathematik, TU Chemnitz, 09107 Chemnitz, FRG, 1997. Available from `http://www.tu-chemnitz.de/sfb393/sfb97pr.html`.

[41] T. Penzl. *Numerische Lösung großer Lyapunov-Gleichungen.* Logos–Verlag, Berlin, Germany, 1998. Dissertation, Fakultät für Mathematik, TU Chemnitz, 1998.

[42] T. Penzl. Algorithms for model reduction of large dynamical systems. Technical Report SFB393/99-40, Sonderforschungsbereich 393 *Numerische Simulation auf massiv parallelen Rechnern*, TU Chemnitz, 09107 Chemnitz, FRG, 1999. Available from `http://www.tu-chemnitz.de/sfb393/sfb99pr.html`.

[43] T. Penzl. A cyclic low rank Smith method for large sparse Lyapunov equations. *SIAM J. Sci. Comput.*, 21(4):1401–1418, 2000.

[44] J. Roberts. Linear model reduction and solution of the algebraic Riccati equation by use of the sign function. *Internat. J. Control*, 32:677–687, 1980. (Reprint of Technical Report No. TR-13, CUED/B-Control, Cambridge University, Engineering Department, 1971).

[45] I. Rosen and C. Wang. A multi–level technique for the approximate solution of operator Lyapunov and algebraic Riccati equations. *SIAM J. Numer. Anal.*, 32(2):514–541, 1995.

[46] V. Sima. *Algorithms for Linear-Quadratic Optimization*, volume 200 of *Pure and Applied Mathematics*. Marcel Dekker, Inc., New York, NY, 1996.

[47] R. Smith. Matrix equation $XA + BX = C$. *SIAM J. Appl. Math.*, 16(1):198–201, 1968.

[48] T. Stykel. Gramian-based model reduction for descriptor systems. *Math. Control Signals Systems*, 16(4):297–319, 2004.

[49] A. Varga. Model reduction software in the SLICOT library. In B. Datta, editor, *Applied and Computational Control, Signals, and Circuits*, volume 629 of *The Kluwer International Series in Engineering and Computer Science*, pages 239–282. Kluwer Academic Publishers, Boston, MA, 2001.

[50] E. Wachspress. Iterative solution of the Lyapunov matrix equation. *Appl. Math. Letters*, 107:87–90, 1988.