Low Rank Solution of Data-Sparse Sylvester Equations*

Ulrike Baur†

October 14, 2005

Abstract

In this paper a method for solving large-scale Sylvester equations is presented. The method is based on the sign function iteration and is particularly effective for Sylvester equations with factorized right-hand side. In this case, the solution will be computed in factored form as it is for instance required in model reduction. The hierarchical matrix format and the corresponding formatted arithmetic is integrated in the iteration scheme to make the method feasible for large-scale computations.

1 Introduction

This paper is concerned with the numerical solution of linear matrix equations of the following form:

$$AX + XB + W = 0, \tag{1}$$

with $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times m}$, $W \in \mathbb{R}^{n\times m}$ and a matrix X of $n \times m$ unknowns. Equations of this type are called *Sylvester equations*. We get an equivalent representation of (1) by using the Kronecker product and by introducing the $vec-$operator,

$$vec : \mathbb{R}^{n\times m} \to \mathbb{R}^{n\cdot m} : W \to [w_{11}, \ldots, w_{n1}, w_{12}, \ldots, w_{n2}, \ldots, w_{nm}]^T :$$

$$(I_m \otimes A + B^T \otimes I_n)\, vec(X) = -vec(W), \tag{2}$$

where I denotes the identity matrix of suitable size. This vectorized representation immediately leads to some first solvability conditions. The coefficient matrix in (2) is regular if and only if the spectra of A and $-B$ are disjoint. This, in turn, is equivalent to the existence and uniqueness of the solution X of (1) [29]. Furthermore, if the two spectra are separated by a line, we get an explicit solution formula [28]:

$$X = \int_0^\infty e^{At} W e^{Bt} dt.$$

*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

For stable Sylvester equations, where the spectra of A and B are both contained in the open left half of the complex plane, these conditions are clearly fulfilled. For the rest of this paper, we will assume stability of the Sylvester equation under consideration.

We are interested in the numerical solution of Sylvester equations, where the dimensions n and m are large. Sylvester equations with this property appear in a wide range of practically relevant applications. For instance, circuit simulation and the spatial discretization of time-dependent partial differential equations result in very large linear dynamical systems of order about $\mathcal{O}(10^5)$. For reducing the dimension of such a system, several model reduction techniques are proposed, see [7] for an overview. The widely used balanced truncation method [31] requires the solution of two Lyapunov equations. These matrix equations are a special symmetric variant of Sylvester equations with $B = A^T$ and symmetric W. A slightly modified model reduction method, the cross-Gramian approach, is based on the solution of one Sylvester equation with $B = A$ [15]. As explained above, the matrix equations arising in model reduction methods are typically large-scale. So we are interested in deriving solvers which are adapted to large-scale computation.

There are several approaches to the numerical solution of Sylvester equations, which can be subdivided into direct and iterative methods. Direct approaches transform the coefficient matrices A and B to Schur [4] or Hessenberg form [14, 16] and solve the resulting linear systems by a backsubstitution process. If we assume that the size of the Sylvester equation is dominated by n, these direct methods are of complexity $\mathcal{O}(n^3)$ and have storage requirements of order $\mathcal{O}(n^2)$ [17, page 367]. Therefore, they are restricted to problems of smaller sizes.

There are also several iterative schemes available, see e.g. [21, 27, 34]. We will focus on the sign function method, published first in 1971 by Roberts [33], which will be described in more detail in Section 2. We will use a special variant of this iteration scheme to compute the solution in factored form $X = YZ$, as proposed in [6], based on a partitioning of the original sign function method. This method is of particular interest in large-scale computations if the solution X has low rank, $\text{rank}(X) \ll n, m$, or at least low numerical rank. In the first case we obtain full-rank factors $Y \in \mathbb{R}^{n\times\text{rank}(X)}$, $Z \in \mathbb{R}^{\text{rank}(X)\times m}$ of X. The latter case is of particular relevance; in many large-scale applications it can be observed that the eigenvalues of X decay rapidly, see e.g. [2, 19, 32]. Then, the memory requirements can be considerably reduced by computing low-rank approximations to the full-rank factors directly. The modified sign function method for computing approximate solution factors is described in Section 3.

In this paper we will propose a new method based on the sign function method for computing low-rank factors of the solution. Despite the low memory requirements for the solution factors, the modified sign function iteration in partitioned form still needs $\mathcal{O}(n^2)$ storage. To make it applicable for larger problems, say $n = \mathcal{O}(n^6)$, we approximate the large-scale matrices A and B during the iteration in a data-sparse format, as so called hierarchical matrices ($\mathcal{H}$-matrices). The $\mathcal{H}$-matrix format is described, e.g., in [18, 20, 23, 24]; it allows data-sparse approximation for a wide, practically relevant class of matrices, which, e.g., arise from boundary element or finite element methods. In [22], Grasedyck, Hackbusch and Khoromskij combine the hierarchical matrix ($\mathcal{H}$-matrix) format with the sign function method for solving algebraic Riccati equations (AREs). The method computes the solution of an ARE in $\mathcal{H}$-matrix format with linear-polylogarithmic complexity. It can be adapted rather directly to the solution of a Sylvester equation, but since we are interested in approximating low rank factorizations of the solution, we propose a new $\mathcal{H}$-matrix arithmetic based iteration scheme. This approach is related to a new algorithm for the solution of Lyapunov equations [5]. In Section 4.1 we give a short introduction in the $\mathcal{H}$-matrix format and the corresponding formatted arithmetic. In Section 4.2 the new algorithm is presented which integrates the $\mathcal{H}$-matrix format and arithmetic in the partitioned iteration scheme of Section 3. Several numerical experiments demonstrate the performance of the new algoritm in Section 5.

2 The Sign Function Method

Consider the square matrix $Z \in \mathbb{R}^{n\times n}$ in Jordan canonical form

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

where the upper block belongs to the eigenvalue of Z with positive real part and $J_{n-\ell}^-$ contains the Jordan blocks belonging to the other eigenvalues. The *matrix sign function* of a matrix Z with no eigenvalues on the imaginary axis is defined as follows:

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

By applying a Newton iteration to the solution of $Z^2 - I_n = 0$:

$$\begin{aligned} Z_0 &\leftarrow Z, \\ Z_{k+1} &\leftarrow \frac{1}{2}(Z_k + Z_k^{-1}), \quad k = 0, 1, 2, \ldots, \end{aligned}$$

we get $\operatorname{sign}(Z) = \lim_{k\to\infty} Z_k$.

We will make use of some special decoupling property of the solution X of the Sylvester equation (1) for computing the matrix sign function of a special matrix Z. We consider the following block upper triangular matrix Z defined by the coefficients of (1):

$$Z = \begin{bmatrix} A & W \\ 0 & -B \end{bmatrix},$$

and a similarity transformation

$$T = \begin{bmatrix} I_n & X \\ 0 & I_m \end{bmatrix}.$$

Then Z is block diagonalized by T,

$$T^{-1} Z T = \begin{bmatrix} A & AX + XB + W \\ 0 & -B \end{bmatrix} = \begin{bmatrix} A & 0 \\ 0 & -B \end{bmatrix},$$

and the matrix sign function gives an expression for the solution of a Sylvester equation:

$$\operatorname{sign}(Z) = T \operatorname{sign}\left(\begin{bmatrix} A & 0 \\ 0 & -B \end{bmatrix}\right) T^{-1} = T \begin{bmatrix} -I_n & 0 \\ 0 & I_m \end{bmatrix} T^{-1} = \begin{bmatrix} -I_n & 2X \\ 0 & I_m \end{bmatrix}.$$

By applying the Newton iteration to Z, an iterative scheme for computing the solution of the Sylvester equation (1) is obtained:

$$\begin{aligned} Z_0 &\leftarrow Z, \\ Z_{k+1} &\leftarrow \frac{1}{2}(Z_k + Z_k^{-1}) \\ &= \begin{bmatrix} \frac{1}{2}(A_k + A_k^{-1}) & \frac{1}{2}(W_k + A_k^{-1} W_k B_k^{-1}) \\ 0 & -\frac{1}{2}(B_k + B_k^{-1}) \end{bmatrix}, \quad k = 0, 1, 2, \ldots. \end{aligned} \tag{3}$$

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

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

as described in [33]. Since $\lim_{k \to \infty} A_k = -I_n$ and $\lim_{k \to \infty} B_k = -I_m$, we get a simple stopping criterion for the iteration:

$$\max_k \{ \|A_k + I_n\|, \|B_k + I_m\| \} \leq \text{tol},$$

with a user-defined tolerance tol. By an appropriate choice of norm and tolerance and by performing two additional iteration steps as proposed in [8], the required accuracy is reached in general owing to the quadratic convergence of the Newton iteration. To overcome slow initial convergence, some of the first 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 suitably chosen parameters. Several choices for such parameters can be found in, e.g., [3, 13, 25]. We will use a problem adapted variant of the *optimal norm scaling* as suggested in [10]:

$$c_k = \sqrt{\frac{\|Z_k^{-1}\|_2}{\|Z_k\|_2}}.$$

3 Factorized Solution of the Sylvester Equation

Many practical applications lead to a Sylvester equation

$$AX + XB + FG = 0, \tag{4}$$

with the constant term in factored form, $F \in \mathbb{R}^{n \times p}$, $G \in \mathbb{R}^{p \times m}$ and A, B stable. For the construction of a well-suited algorithm for the solution of equations of this type, we will make use of the following observation. Often, for large-scale Sylvester equations, the solution X has a low numerical rank. In [19] it is shown that the singular values of X decay exponentially if the right-hand side is of low rank and the spectra of A and $-B$ are separated by a line. There are several other papers which present eigenvalue decay bounds for Sylvester equations of a certain structure, e.g. [2, 32]. Based on this observation, we modify the iteration scheme as proposed in [6] for computing the solution X in factored form, $X = YZ$, with $Y \in \mathbb{R}^{n\times \mathrm{rank}(X)}$, $Z \in \mathbb{R}^{\mathrm{rank}(X)\times m}$. The storage requirements for X are reduced from $\mathcal{O}(n \times m)$ to $\mathcal{O}((n+m) \times \mathrm{rank}(X))$. We rewrite the Newton iteration (3) with $A_0 \leftarrow A$, $B_0 \leftarrow B$, $F_0 \leftarrow F$, $G_0 \leftarrow G$:

$$\begin{aligned}
A_{k+1} &\leftarrow \frac{1}{2}(A_k + A_k^{-1}), \\
B_{k+1} &\leftarrow \frac{1}{2}(B_k + B_k^{-1}), \\
F_{k+1} &\leftarrow \frac{1}{\sqrt{2}}\left[\, F_k,\ A_k^{-1}F_k \,\right], \\
G_{k+1} &\leftarrow \frac{1}{\sqrt{2}}\left[\begin{array}{c} G_k \\ G_kB_k^{-1} \end{array}\right]
\end{aligned} \tag{5}$$

and get $Y = \frac{1}{\sqrt{2}} \lim_{k\to\infty} F_k$ and $Z = \frac{1}{\sqrt{2}} \lim_{k\to\infty} G_k$ as solution factors of (4). This iteration scheme is less expensive during the first iteration steps, if we assume that $p \ll n, m$. In the course of the iteration, this advantage gets lost as p_k, the number of columns of the F-iterates and the number of rows of the G-iterates, is doubled in each step. As mentioned before, we expect that the solution has low numerical rank; it can therefore be expected that the iterates are also of low numerical rank. To exploit this property and to avoid the exponential growth of the columns and rows, we apply a rank-revealing QR factorization (RRQR) [17] to F_{k+1} and G_{k+1} in each iteration step.

For a given matrix M with singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq 0$ the RRQR factorization is defined as

$$M = QR\Pi = Q \left[\begin{array}{cc} R_{11} & R_{12} \\ 0 & R_{22} \end{array}\right] \Pi,$$

with a permutation matrix Π, an orthonormal matrix Q and an $r \times r$ matrix R_{11}. The order r of R_{11} denotes the numerical rank of the matrix M for a given threshold τ. The numerical rank r is defined by the smallest singular value which satisfies $\sigma_r \leq \sigma_1 \cdot \tau$. For an RRQR, the condition number κ of R_{11} can be bounded by $1/\tau$,

$$\kappa(R_{11}) := \|R_{11}\|_2 \cdot \|R_{11}^{-1}\|_2 \approx \sigma_1/\sigma_r \leq 1/\tau.$$

We compress the rows of M by only considering entries in the upper part of the matrix R, that is the well-conditioned part of M.

In our iteration scheme, the RRQR factorization is integrated as follows:

$$
\begin{aligned}
F_{k+1}G_{k+1} &= \frac{1}{2}\left[\, F_k,\, A_k^{-1}F_k \,\right]\left[\begin{array}{c} G_k \\ G_k B_k^{-1} \end{array}\right] \\
&= \frac{1}{2}\left[\, F_k,\, A_k^{-1}F_k \,\right] U R\, \Pi_G \\
&= \frac{1}{2}\underbrace{\left[\, F_k,\, A_k^{-1}F_k \,\right] U}_{V T\, \Pi_F}\left[\begin{array}{cc} R_{11} & R_{12} \\ 0 & R_{22} \end{array}\right]\Pi_G \\
&= \frac{1}{2}V\left[\begin{array}{cc} T_{11} & T_{12} \\ 0 & T_{22} \end{array}\right]\Pi_F\left[\begin{array}{cc} R_{11} & R_{12} \\ 0 & R_{22} \end{array}\right]\Pi_G.
\end{aligned} \tag{6}
$$

Π_F and Π_G are permutation matrices, U and V have orthonormal columns, $R_{11} \in \mathbb{R}^{r\times r}$, $T_{11} \in \mathbb{R}^{t\times t}$. The numerical rank of G_{k+1} is denoted by r, t is the numerical rank of F_{k+1}, both with respect to a given threshold τ. We get approximate iterates $\tilde{F}_{k+1}$ and $\tilde{G}_{k+1}$ by truncating the matrices T and R, by a partitioning of $[\, T_{11},\, T_{12}\,]\,\Pi_F$ to $\left[\, \tilde{T}_{11},\, \tilde{T}_{12} \,\right]$ with $\tilde{T}_{11} \in \mathbb{R}^{t\times r}$ to adapt the matrix dimensions of the two factors and by a further partitioning of the orthonormal matrix $V = [\, V_1,\, V_2\,]$ with $V_1 \in \mathbb{R}^{n\times t}$:

$$
\begin{aligned}
\tilde{G}_{k+1} &:= [\, R_{11},\, R_{12}\,]\,\Pi_G, \\
\left[\, \tilde{T}_{11},\, \tilde{T}_{12} \,\right] &:= [\, T_{11},\, T_{12}\,]\,\Pi_F, \\
\tilde{F}_{k+1} &:= V_1 \tilde{T}_{11}.
\end{aligned} \tag{7}
$$

The iterates $\tilde{F}_{k+1}$ and $\tilde{G}_{k+1}$ have a reduced number of columns and rows, respectively, $p_{k+1} := r$ instead of $2p_k$, and we obtain approximate solution factors $\tilde{Y}$ and $\tilde{Z}$ by

$$
\tilde{Y} = \frac{1}{\sqrt{2}}\lim_{k\to\infty}\tilde{F}_k, \quad \tilde{Z} = \frac{1}{\sqrt{2}}\lim_{k\to\infty}\tilde{G}_k.
$$

4 $\mathcal{H}$-Matrix Arithmetic based Sign Function Iteration

In the previous section we considered a modified iteration scheme (5) with integrated RRQR factorization (7) for the computation of approximate full-rank solution factors. Despite the low memory requirements for the solution, we still have storage requirements of order $\mathcal{O}(n^2+m^2)$ and $\mathcal{O}(n^3+m^3)$ operations during the Newton iteration for the iterates A_k and B_k. Therefore we will integrate a data-sparse matrix format and the corresponding approximate arithmetic in our iteration scheme to make it feasible for large-scale computations. In the following, we will give a short introduction into this matrix format.

4.1 Short Introduction into $\mathcal{H}$-Matrix Arithmetic

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 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 [18, 20, 23, 24].

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 $M_{|_{r\times s}}$, which is stored in factorized form with low rank k:

$$M_{|_{r\times s}} = AB^T, \quad A \in \mathbb{R}^{t\times k},\, B \in \mathbb{R}^{s\times k}. \tag{8}$$

A matrix $M \in \mathbb{R}^{n\times n}$ can be stored 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., [20] 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:

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

Here, $\tilde{A}^{-1}$ denotes the approximate inverse of A which is obtained by performing block Gaussian elimination on A with formatted addition and multiplication. In some situations it is recommended to compute the inverse V of a matrix A using an approximate $\mathcal{H}$-LU factorization $A \approx L_{\mathcal{H}}U_{\mathcal{H}}$ followed by an $\mathcal{H}$-forward ($L_{\mathcal{H}}W = (I)_{\mathcal{H}}$) and $\mathcal{H}$- backward substitution ($U_{\mathcal{H}}V = W$), see [20] for more details.

In [22], the sign function method for solving the more general algebraic Riccati equation is combined with a data-sparse matrix representation and a corresponding approximate arithmetic. A method for the factorized solution of Lyapunov equations based on the hierarchical matrix arithmetic is proposed in [5]. Our approach also makes use of this $\mathcal{H}$-matrix structure, as described in the next section.

4.2 Algorithm

We consider the sign function iteration in the partitioned form (5) to compute full-rank factors Y and Z of the solution X of the Sylvester equation (4).

Even if the system matrices A and B in (4) are sparse, resulting, e.g., from a finite element discretizations of elliptic partial differential operators, a large amount of memory is required during the Newton iteration caused by fill-in during the matrix inversion. To avoid this effect, the large-scale iterates A_k and B_k are approximated in the data-sparse $\mathcal{H}$-matrix format and the hierarchical matrix arithmetic is used to reduce the computational cost in these iteration parts (compare with Section 4.1). Instead of the formatted matrix inversion we compute an LU decomposition of the matrices A_k and B_k and an $\mathcal{H}$-based forward/backward substitution to obtain approximate inverses V_A and V_B. This has the advantage of lower storage requirements since approximate inversion takes roughly three times the workspace occupied by the original matrix.

The matrices F_k and G_k, which yield the solution factors at the end of the iteration, are stored in the usual "full" format. In these iteration parts arithmetic operations from standard linear algebra packages such as LAPACK [1] and BLAS [30] can be used. We integrate the RRQR factorization in the iteration scheme as described in Section 3 to limit the increasing number of columns and rows of the two solution factors. Since $\lim_{k\to\infty} A_k = -I_n$ and $\lim_{k\to\infty} B_k = -I_m$, as it was seen in Section 2, it is advised to choose

$$\max_k\{\|A_k + I_n\|_2, \|B_k + I_m\|_2\} \leq \text{tol}, \tag{9}$$

with a user-defined tolerance tol, as stopping criterion for the iteration, which is easy to check.

We introduce scaling to accelerate the initial convergence. Due to error amplification during the sign function iteration with formatted arithmetic, scaling is used only in the first iteration step as in [18]. We will use a problem adapted variant of the *optimal norm scaling* to balance the norms of the summands in line 10 and line 15 of Algorithm 1 as suggested in [10],

$$c = \sqrt{\frac{\|Z_0^{-1}\|_2}{\|Z_0\|_2}},$$

with the norm approximations

$$\|Z_0\|_2 \approx \sqrt{\|A_0\|_2\|B_0\|_2}, \quad \|Z_0^{-1}\|_2 \approx \sqrt{\|V_A\|_2\|V_B\|_2}.$$

In the partitioned iteration scheme of Algorithm 1 scaling is integrated in the following way:

$$\begin{aligned}
A_1 &\leftarrow \frac{1}{2}(cA_0 \oplus \frac{1}{c}V_A),\\
B_1 &\leftarrow \frac{1}{2}(cB_0 \oplus \frac{1}{c}V_B),\\
F_1 &\leftarrow \frac{1}{\sqrt{2}}\left[\begin{array}{cc} \sqrt{c}F_0 & \frac{1}{\sqrt{c}}V_AF_0 \end{array}\right],\\
G_1 &\leftarrow \frac{1}{\sqrt{2}}\left[\begin{array}{c} \sqrt{c}G_0 \\ \frac{1}{\sqrt{c}}G_0V_B \end{array}\right].
\end{aligned}$$

5 Numerical Experiments

All numerical experiments were performed on an SGI Altix 3700 (32 Itanium II processors, 1300 MHz, 64 GBytes RAM). We made use of the LAPACK and

Algorithm 1 Calculate full-rank factors Y, Z of X for $AX + XB + FG = 0$

INPUT: $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times m}$, $F \in \mathbb{R}^{n\times p}$, $G \in \mathbb{R}^{p\times m}$, tol, τ

OUTPUT: Approximations Y and Z to full-rank factors of the solution X.

$A_0 \leftarrow (A)_{\mathcal{H}}$
$B_0 \leftarrow (B)_{\mathcal{H}}$
$F_0 \leftarrow F$
$G_0 \leftarrow G$
$k = 0$
while $\max\{\|A_k + I_n\|, \|B_k + I_m\|\} > \text{tol}$ **do**
    $[L, U] \leftarrow LU_{\mathcal{H}}(A_k)$
    Solve $LW = (I_n)_{\mathcal{H}}$ by $\mathcal{H}$-forward substitution.
    Solve $UV_A = W$ by $\mathcal{H}$-back substitution.
    $A_{k+1} \leftarrow \frac{1}{2}(A_k \oplus V_A)$
    $F_{k+1} \leftarrow \frac{1}{\sqrt{2}} \begin{bmatrix} F_k & V_A F_k \end{bmatrix}$
    $[L, U] \leftarrow LU_{\mathcal{H}}(B_k)$
    Solve $LW = (I_m)_{\mathcal{H}}$ by $\mathcal{H}$-forward substitution.
    Solve $UV_B = W$ by $\mathcal{H}$-back substitution.
    $B_{k+1} \leftarrow \frac{1}{2}(B_k \oplus V_B)$
    $G_{k+1} \leftarrow \frac{1}{\sqrt{2}} \begin{bmatrix} G_k \\ G_k V_B \end{bmatrix}$
    Compress rows of G_{k+1} to r using a RRQR with threshold τ (see (6), (7)).
    Compress rows of $F_{k+1}U$ using a RRQR with threshold τ (see (6), (7)).
    Cut off columns of $F_{k+1}U$ to r (see (7)).
    $k = k + 1$
end while
$Y \leftarrow \frac{1}{\sqrt{2}} F_k$, $Z \leftarrow \frac{1}{\sqrt{2}} G_k$

BLAS libraries for performing the standard dense matrix operations and include the routine DGEQPX of the RRQR library [11] for computing the RRQR factorization. For the $\mathcal{H}$-matrix approximation we employ HLib 1.2 [12]. We use the adaptive rank choice (see [18]) instead of a given fixed rank. The truncation operator of the approximate $\mathcal{H}$-matrix arithmetic is then changed in the following way:

$$\mathcal{T}_\epsilon(A) = \operatorname{argmin}\left\{ \operatorname{rank}(R) \;\middle|\; \frac{\|R - A\|_2}{\|A\|_2} \leq \epsilon \right\}, \tag{10}$$

where the parameter ϵ determines the desired accuracy in each matrix block. For the stopping criterion (9) we take the threshold tol$= 1.e - 04$.

Example 5.1. In this example we consider a control problem for the two-dimensional heat equation as described in [22]. We discretized the partial differential equation with linear finite elements and n inner grid points. This results in a linear time-invariant system

$$\begin{array}{rcll} \dot{x}(t) & = & Ax(t) + Fu(t), & t > 0, \quad x(0) = x^0, \\ y(t) & = & Gx(t), & t \geq 0, \end{array} \tag{11}$$

with a stable matrix $A \in \mathbb{R}^{n\times n}$ and $F, G^T \in \mathbb{R}^{n\times 1}$. Thus, we have system with a single input and a single output (SISO). For reducing the oder n of

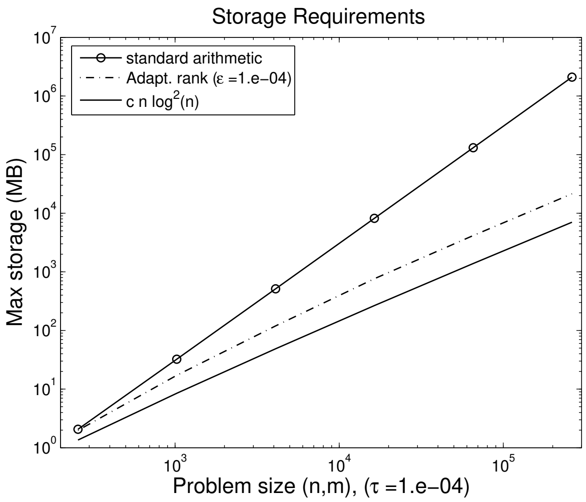

Figure 1: Maximal storage requirements in logarithmic scale for Algorithm 1 in $\mathcal{H}$-matrix arithmetic and in standard arithmetic compared to an $\mathcal{O}(n \log^2 n)$ reference line.

this system we consider a variant of the classical balanced truncation model reduction approach. This method requires the solution of a special Sylvester equation

$$AX + XA + FG = 0, \tag{12}$$

where the solution $X \in \mathbb{R}^{n \times n}$ is called the cross-Gramian associated with the system (11). In our example we test the iteration scheme for the cross-Gramian of a SISO system. We vary the problem size from $n = 256$ to $n = 262\,144$ and choose fixed values for the numerical rank decision in the RRQR factorization: $\tau = 1.e-04$ and as approximation error in the adaptive rank choice of the $\mathcal{H}$-matrix arithmetic: $\epsilon = 1.e-04$. With our algorithm we compute the approximate solutions factors Y and Z of the cross-Gramian X. We compare the solution from the $\mathcal{H}$-matrix arithmetic based sign function iteration with the solution computed with the primary iteration scheme in Section 3. In the latter scheme all matrices are stored in the usual "full" format and the matrix operations are performed in standard arithmetic. Due to the large memory requirements (see Figure 1) these solutions are only computed up to a problem size of $n = 4096$, larger results are extrapolated in the two figures or omitted in Table 1. The results of this computation are depicted in columns with column heading "full". In Figures 1 and 2 it is seen that the storage requirements as well as the computational time for the algorithm in $\mathcal{H}$-matrix arithmetic exhibit almost linear growth. The ranks of the factors of the cross-Gramian and their accuracy are plotted in Table 1. As a measure of accuracy we consider the relative residual

$$\frac{\|AX + XA + FG\|_2}{2\|A\|_2\|X\|_2 + \|F\|_2\|G\|_2},$$

which could be considered as the backward error for an approximate solution of the Sylvester equation (up to an amplification factor described in [26, Chapter

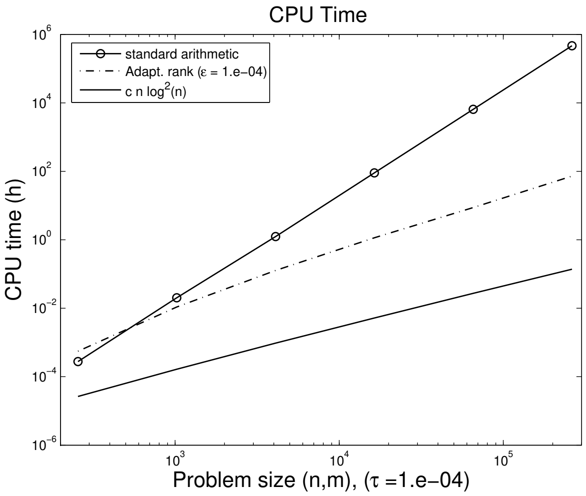

Figure 2: CPU time in logarithmic scale for Algorithm 1 in $\mathcal{H}$-matrix arithmetic and in standard arithmetic compared to an $\mathcal{O}(n \log^2 n)$ reference line.

15]). It is computed up to a problem size of $n = 16\,384$ due to storage requirements and seems to be bounded above for increasing problem size. For smaller problems the relative errors $\frac{\|X_* - X\|_2}{\|X_*\|_2}$ are computed with the reference solution X_* in "full" format and with standard arithmetic.

n	# iter.	r		time[sec]		rel. residual		rel. error
		$\mathcal{H}$	full	$\mathcal{H}$	full	$\mathcal{H}$	full	
256	10	12	12	2	1	9.3e-08	3.2e-08	7.43e-16
1024	11	13	13	39	73	1.1e-06	1.3e-08	3.92e-07
4096	12	14	14	459	4484	2.6e-06	7.0e-09	1.58e-06
16384	13	15	-	4124	-	5.4e-06	-	-
65536	14	17	-	31454	-	-	-	-
262144	15	17	-	261263	-	-	-	-

Table 1: Accuracy and rank r of the computed solution factors of Algorithm 1 for different problem sizes.

It should be noted that the largest Sylvester 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 approximate full-rank factors $Y, Z^T \in \mathbb{R}^{n \times 17}$ and therefore need 8.5 MB memory to store the solution instead of 64 GB for the explicit solution X. □

Example 5.2. We tested the new algorithm with matrices A, B, F, G stemming from a semi-discretization of the same control problem for the two-dimensional heat equation as in Example 5.1. For the space discretization we consider linear finite element ansatz spaces of different sizes n and m which results in different matrix dimensions of the square matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{m\times m}$ and of $F \in \mathbb{R}^n$, $G^T \in \mathbb{R}^m$. For a fixed size $n = 4096$ we vary the number of grid points m from 256 to 65 536. We take the same stopping criterion as in Example 5.1 and also the same fixed choice of parameter values: $\epsilon = 1.e-04$, $\tau = 1.e-04$.

n	m	#	r		time[sec]		rel. residuum		rel error
			$\mathcal{H}$	full	$\mathcal{H}$	full	$\mathcal{H}$	full	
4096	256	13	11	11	383	2419	4.4e-06	2.0e-06	1.53e-06
4096	1024	13	13	13	268	2475	3.1e-06	3.4e-07	1.53e-06
4096	4096	12	14	14	459	4484	2.6e-06	7.0e-09	1.58e-06
4096	16384	14	15	-	2832	-	6.0e-06	-	-
4096	65536	15	17	-	56346	-	-	-	-

Table 2: Accuracy and rank r of the computed solution factors of Algorithm 1 for different problem sizes in m.

Again, we observe high accuracy in the solution factors computed with the algorithm in $\mathcal{H}$-matrix arithmetic. The relative residual as well as the relative error are observed to remain bounded above for increasing problem size. The execution time for the algorithm in $\mathcal{H}$-matrix arithmetic is considerably lower than the time needed by the algorithm in standard dense format. □

Example 5.3. Now we fix the problem size for the system described in Example 5.2 by $n = m = 4096$. We test various parameter combinations of ϵ and τ, where τ is the threshold for the numerical rank decision in the rank-revealing QR factorization, ϵ is the parameter for the adaptive choice of rank in an $\mathcal{H}$-matrix subblock. Previous results for the solution of Lyapunov equations and an error analysis in [5] suggest that no accuracy improvements can be expected by choosing the parameter τ smaller than ϵ; we therefore did not consider this case. For the stopping criterion we take the threshold tol$= 1.e-04$.

ϵ	τ	#	r		time[sec]		rel. residuum		rel error
			$\mathcal{H}$	full	$\mathcal{H}$	full	$\mathcal{H}$	full	
1.e-04	1.e-04	12	14	14	459	4563	2.6e-06	7.0e-09	1.58e-06
1.e-08	1.e-04	12	14	14	1805	4517	7.0e-09	7.0e-09	5.22e-11
1.e-08	1.e-08	12	30	30	1810	4492	2.0e-10	8.4e-16	5.22e-11
1.e-16	1.e-08	12	30	30	8703	4519	8.8e-16	8.4e-16	1.22e-15
1.e-16	1.e-16	12	69	75	8792	4335	6.4e-16	7.0e-16	1.22e-15

Table 3: Accuracy and rank r of the computed solution factors of Algorithm 1 for different parameter variations and $n = 4096$, $m = 4096$.

The results of the parameter variations show the expected behavior, we have increasing accuracy as ϵ gets smaller. A choice of $\epsilon = 1.e-16$ results in large computational time and large storage requirements since the local ranks in the matrix blocks have to be very large to fulfill the accuracy condition (10). Therefore the benefits of the $\mathcal{H}$-matrix approach from low-rank approximations of the submatrices get lost. The storage requirements might get even larger than in "full" format, compare (8), and it is consequently recommended to choose the parameter ϵ of moderate size. The dimension of the solution factors increases with τ getting smaller which has impact on the accuracy for the results in standard arithmetic. A decreasing of τ did not considerably improve the accuracy in the $\mathcal{H}$-matrix computation. This observation fits to a criterion presented in [9, page 21], which suggests to choose the RRQR threshold τ of the same order as the square root of the desired accuracy. $\square$

6 Conclusions

In this paper a new algorithm for the solution of Sylvester equations in factorized form is presented. This algorithm computes the factorized solution of Sylvester equations arising from FEM/BEM discretizations of elliptic partial differential operators. With the $\mathcal{H}$-matrix based sign function approach we have significant savings in computational time and memory requirements during the iteration and due to the modified iteration scheme for computing the solution factors directly we have additional savings in memory requirement for the approximate full-rank solution factors if the solution of the Sylvester equation has low numerical rank. Therefore the algorithm is well-suited for model reduction based on balanced truncation by using the cross-Gramian where we have to solve large-scale Sylvester equations with factorized right-hand side.

Acknowledgements

I would like to thank Peter Benner for support, useful discussions and suggestions and Daniel Kressner for giving useful comments. This work was supported by the DFG Research Center MATHEON "Mathematics for key technologies" in Berlin.

References

[1] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. SIAM, Philadelphia, PA, third edition, 1999.

[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] 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.

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

[5] U. Baur and P. Benner. Factorized solution of Lyapunov equations based on hierarchical matrix arithmetic. Preprint #161, DFG Research Center "Mathematics for Key Technologies", Berlin, FRG, `http://www.math.tu-berlin.de/DFG-Forschungszentrum`, Oct. 2004.

[6] P. Benner. Factorized solution of Sylvester equations with applications in control. In *Proc. Intl. Symp. Math. Theory Networks and Syst. MTNS 2004*, `http://www.mtns2004.be`, 2004.

[7] P. Benner, V. Mehrmann, and D. Sorensen, editors. *Dimension Reduction of Large-Scale Systems*, volume 45 of *Lecture Notes in Computational Science and Engineering*. Springer-Verlag, Berlin/Heidelberg, Germany, 2005.

[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 and E. Quintana-Ortí. Model reduction based on spectral projection methods. In P. Benner, V. Mehrmann, and D. Sorensen, editors, *Dimension Reduction of Large-Scale Systems*, volume 45 of *Lecture Notes in Computational Science and Engineering*, pages 5–45. Springer-Verlag, Berlin/Heidelberg, Germany, 2005.

[10] P. Benner, E. Quintana-Ortí, and G. Quintana-Ortí. Solving stable Sylvester equations via rational iterative schemes. *J. Sci. Comp.*, to appear.

[11] C. Bischof and G. Quintana-Ortí. Algorithm 782: codes for rank-revealing QR factorizations of dense matrices. *ACM Trans. Math. Software*, 24(2):254–257, 1998.

[12] S. Börm, L. Grasedyck, and W. Hackbusch. HLib 1.2, 2004. Available from `http://www.hlib.org`.

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

[14] W. Enright. Improving the efficiency of matrix operations in the numerical solution of stiff ordinary differential equations. *ACM Trans. Math. Softw.*, 4:127–136, 1978.

[15] K. Fernando and H. Nicholson. On the structure of balanced and other principal representations of SISO systems. *IEEE Trans. Automat. Control*, 28(2):228–231, 1983.

[16] G. H. Golub, S. Nash, and C. F. Van Loan. A Hessenberg–Schur method for the problem $AX + XB = C$. *IEEE Trans. Automat. Control*, AC-24:909–913, 1979.

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

[18] 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`.

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

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

[21] 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.

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

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

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

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

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

[27] D. Y. Hu and L. Reichel. Krylov-subspace methods for the Sylvester equation. *Linear Algebra Appl.*, 172:283–313, 1992.

[28] P. Lancaster. Explicit solutions of linear matrix equation. *SIAM Rev.*, 12:544–566, 1970.

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

[30] C. Lawson, R. Hanson, D. Kincaid, and F. Krogh. Basic linear algebra subprograms for FORTRAN usage. *ACM Trans. Math. Software*, 5:303–323, 1979.

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

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

[33] J. D. 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).

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