FASSBENDER, H.; BENNER, P.

Computing roots of matrix products

The problem of computing a kth root of a matrix product $W = \prod_{i=1}^{k} A_i$ is considered. The explicit computation of W may produce a highly inaccurate result due to rounding errors, such that the computed root $\widehat{W}^{\frac{1}{k}}$ is far from the actual root $W^{\frac{1}{k}}$. An algorithm for computing the kth root of W is presented which avoids the explicit computation of W by employing the periodic Schur decomposition and therefore yields better accuracy in the computed root $\widehat{W}^{\frac{1}{k}}$.

1. Introduction

Computing the square root of the product of two matrices can be used in model reduction methods based on the cross-Gramians; see [1] and the references therein. This lead us to consider the more general problem of computing the kth root of a matrix product $W = A_1A_2\cdots A_{k-1}A_k$ where $A_i \in \mathbb{R}^{n\times n}$. That is, a matrix $W^{\frac{1}{k}}$ is sought such that $W = (W^{\frac{1}{k}})^k$. A kth root of a matrix may not exist. For example, it is easy to verify that the matrix $\left[\begin{smallmatrix} 0 & 1 \\ 0 & 0 \end{smallmatrix}\right]$ has no square root. Here we will not discuss existence results any further.

The direct approach for computing the kth root of W involves first the explicit computation of W followed by the computation of its kth root. Already the explicit computation of W may produce a highly inaccurate result due to rounding errors, such that the computed product $\widehat{W}$ is far from the actual matrix W. Then one can not expect that the computed root $\widehat{W}^{\frac{1}{k}}$ is close to the actual root $W^{\frac{1}{k}}$. Hence explicitly forming the matrix W should be avoided if possible. The approach suggested here was inspired by the work in [2]. There a fast and stable method for computing the square root X of a given matrix A is developed. The method is based on the Schur factorization $A = QSQ^H$ and uses a fast recursion to compute the upper triangular square root U of S. Then $X = QUQ^H$ is the desired square root. The fast recursion is obtained by comparing coefficients in the equation $U^2 = S$.

The algorithm for computing the kth root of W sketched in the following avoids the explicit computation of W by employing the periodic Schur decomposition. That is, the real Schur factorization $W = Q_1RQ_1^T$ will be computed implicitly by simultaneously reducing all but the first A_j to upper triangular matrices R_j, A_1 can only be reduced to a quasi-upper triangular matrix R_1. Then the desired kth root is computed by computing the root of the product of the factors R_j. This implies that the computed root will have the same quasi-upper triangular form as R_1. This root is then transformed back to the root of W. But note that even in case a kth root of $Q_1^TWQ_1$ exists, it must not necessarily have the form of R_1.

In order to keep the presentation here simple, the process will only be demonstrated for the case $k = 2$. As the explicit forming of the product matrix W is avoided this approach yields better accuracy in the computed root $\widehat{W}^{\frac{1}{k}}$ than when using the direct approach.

2. First step: The periodic Schur decomposition

In order to avoid explicitly computing the matrix W a two step approach for computing the kth root of W is proposed which implicitly computes the desired root without ever forming the explicit matrix product W. The first step of the algorithm consists of employing the periodic Schur decomposition.

Theorem 1. *Let $A_j \in \mathbb{R}^{n\times n}$, $j = 1,\ldots,k$. There exist orthogonal matrices Q_j, $j = 1,\ldots,k$, such that*

$$R_1 = Q_1A_1Q_2^T, \quad R_2 = Q_2A_1Q_3^T, \quad \ldots, \quad R_{k-1} = Q_{k-1}A_{k-1}Q_k^T, \quad R_k = Q_kA_1Q_1^T,$$

where $R_2,\ldots,R_k$ are upper triangular and R_1 is in quasi-upper triangular form with 1×1 and 2×2 diagonal blocks. Moreover, Q_1 puts W into real Schur form, i.e. $Q_1^TWQ_1 = R_1R_2\cdots R_{k-1}R_k$ is quasi-upper triangular.

Constructive proofs and algorithms for computing the periodic Schur decomposition without explicitly forming the product W can be found in [3] and [4].

3. Second step: The kth root

Here we will assume $k = 2$ for simplicity. Using the periodic Schur decomposition the matrix product

$W = A_1A_2$ is transformed to $\widehat{W} = Q_1^TWQ_1 = (Q_1^TA_1Q_2)(Q_2^TA_2Q_1)$ such that $H = Q_1^TA_1Q_2$ is quasi-upper triangular and $R = Q_2^TA_2Q_1$ is upper triangular. $\widehat{W}$ has the same quasi-triangular form as H. Comparing the coefficients in $\widehat{W} = X^2$ gives us formulae to compute the elements of the square root X of $\widehat{W}$. Then $W^{\frac{1}{2}} = Q_1XQ_1^T$. In order to compute the elements of X we first have to consider the diagonal blocks. In case of a 1×1 block h_{kk} with $h_{k+1,k} = h_{k,k-1} = 0$ we can directly read off $x_{kk} = \sqrt{h_{kk}r_{kk}}$. In case of a 2×2 block $\left[\begin{smallmatrix} h_{kk} & h_{k,k+1} \\ h_{k+1,k} & h_{k+1,k+1} \end{smallmatrix}\right]$ we follow the approach taken in [2]. First orthogonal matrices U and V are computed such that $\widetilde{R} = V^T \left[\begin{smallmatrix} r_{kk} & r_{k,k+1} \\ 0 & r_{k+1,k+1} \end{smallmatrix}\right] U$ is upper triangular, $\widetilde{H} = U^T \left[\begin{smallmatrix} h_{kk} & h_{k,k+1} \\ h_{k+1,k} & h_{k+1,k+1} \end{smallmatrix}\right] V$ and $\widetilde{H}\widetilde{R} = \left[\begin{smallmatrix} a & b \\ c & a \end{smallmatrix}\right]$ with $cb < 0$ (that is, $\widetilde{H}\widetilde{R}$ is in standard real Schur form). Then

$$x_{kk} = x_{k+1,k+1} = \sqrt{\tfrac{1}{2}(a + \sqrt{a^2 - bc})}, \quad x_{k+1,k} = \tfrac{1}{2x_{kk}}c, \quad x_{k,k+1} = \tfrac{1}{2x_{kk}}b.$$

The transformation matrices U and V have to be applied to rows and columns $k, k+1$ of H, R and Q_1 in order to complete this part of the computation. Now the other elements of X can be computed one superdiagonal at a time. Four different cases have to be distinguished depending on the indices of x_{jk}:

1.: the indices j and k are such that neither h_{jj} nor h_{kk} belong to a 2×2 diagonal block, then

$$x_{jk} = \frac{\sum_{\ell=\max(1,j-1)}^{k} h_{j\ell}r_{\ell k} - \sum_{\ell=j+1}^{k-1} x_{j\ell}x_{\ell k}}{x_{jj}+x_{kk}}.$$

2.: index j is such that h_{jj} does not belong to a 2×2 diagonal block, while index k is such that h_{kk} belongs to a 2×2 diagonal block (assume w.l.o.g. that the 2×2 block is given by $\left[\begin{smallmatrix} h_{kk} & h_{k,k+1} \\ h_{k+1,k} & h_{k+1,k+1} \end{smallmatrix}\right]$), then we obtain a 2×2 system of linear equations for x_{jk} and $x_{j,k+1}$ which yields with $s_p = \sum_{\ell=j}^{p} h_{j\ell}r_{\ell p} - \sum_{\ell=j+1}^{k-1} x_{j\ell}x_{\ell p}$

$$x_{j,k+1} = \frac{s_{k+1}(x_{jj}+x_{kk}) - s_k x_{k,k+1}}{(x_{jj}+x_{kk})(x_{jj}+x_{k+1,k+1}) - x_{k,k+1}x_{k+1,k}}, \quad x_{jk} = \frac{s_k - x_{j,k+1}x_{k+1,k}}{x_{jj}+x_{kk}}.$$

3.: index k is such that h_{kk} does not belong to a 2×2 diagonal block, while index j is such that h_{jj} belongs to a 2×2 diagonal block (assume w.l.o.g. that the 2×2 block is given by $\left[\begin{smallmatrix} h_{j-1,j-1} & h_{j-1,j} \\ h_{j,j-1} & h_{jj} \end{smallmatrix}\right]$), then we obtain a 2×2 system of linear equations for x_{jk} and $x_{j-1,k}$ which yields with $s_p = \sum_{\ell=\max(1,j-1)}^{k} h_{p\ell}r_{\ell k} - \sum_{\ell=j+1}^{k-1} x_{p\ell}x_{\ell k}$

$$x_{jk} = \frac{s_j(x_{j-1,j-1}+x_{kk}) - x_{j,j-1}s_{j-1}}{(x_{jj}+x_{kk})(x_{j-1,j-1}+x_{kk}) - x_{j,j-1}x_{j-1,j}}, \quad x_{j-1,k} = \frac{s_{j-1} - x_{j-1,j}x_{jk}}{x_{j-1,j-1}+x_{kk}}.$$

4.: both indices j and k are such that h_{jj} and h_{kk} belong to 2×2 diagonal blocks (assume as before, that h_{jj} and h_{kk} are the $(2,2)$, resp. the $(1,1)$ element of the respective 2×2 block), then the 4 elements of $X_{jk} = \left[\begin{smallmatrix} x_{j-1,k} & x_{j-1,k+1} \\ x_{j,k} & x_{j,k+1} \end{smallmatrix}\right]$ can be computed via solving a 2×2 Sylvester equation

$$\begin{bmatrix} x_{j-1,j-1} & x_{j-1,j} \\ x_{j,j-1} & x_{jj} \end{bmatrix} X_{jk} + X_{jk} \begin{bmatrix} x_{k,k} & x_{k,k+1} \\ x_{k+1,k} & x_{k+1,k+1} \end{bmatrix} = C_{jk},$$

where $C_{jk} = H_{j-1:j,1:k+1}R_{1:k+1,k:k+1} - X_{j-1:j,j+1:k-1}X_{j+1:k-1,k:k+1}$ (employing standard MATLAB notation).

Numerical examples show that the approach presented here for computing the square root of a matrix product $W = A_1A_2$ gives better accuracy than when applying the method proposed in [2] to the product W. Similar to the approach above algorithms for computing the kth root of a product of k matrices can be derived. These formulae are much more involved, e.g., for a 3rd root 2×2 equations of the form $A^2X + AXB + XB^2 = C$ have to be solved. Stability issues and existence of such a kth root need further discussion.

4. References

1 ALDHAHERI, R.W.: Model order reduction via real Schur-form decomposition. Int. J. Control **53** (1991), 709–716.

2 BJÖRCK, Å. AND HAMMARLING, S.: A Schur Method for the Square Root of a Matrix. Lin. Alg. Appl. **52/53** (1983), 127–140.

3 BOJANCZYK, A., GOLUB, G.H. AND VAN DOOREN, P.: The periodic Schur decomposition, algorithms and applications. In Proc. SPIE Conf., **1770** (1992), 31–42.

4 HENCH, J.J. AND LAUB, A.J.: Numerical Solution of the Discrete-Time Periodic Riccati Equations. IEEE Trans. Aut. Cont. **39** (1994), 1197–1210.

Addresses: HEIKE FASSBENDER, Technische Universität München, Fakultät für Mathematik, 80290 München, FRG.
PETER BENNER, Universität Bremen, FB 3, Zentrum für Technomathematik, 28334 Bremen, FRG.