A QR-decomposition of block tridiagonal matrices generated by the block Lanczos process

Thomas Schmelzer
Oxford University Computing Laboratory
Oxford University
Wolfson Building, Parks Road
Oxford, OX1 3UQ United Kingdom
Phone: +44-1865-273855, Fax: +44-1865-273839
E-mail: thoms@comlab.ox.ac.uk

Martin H. Gutknecht
Seminar for Applied Mathematics, ETH Zurich
ETH Zurich
ETH-Zentrum, HG G52.2
CH-8092 Zurich, Switzerland
Phone: +41-1-632-3464, Fax: +41-1-632-1104
E-mail: mhg@sam.math.ethz.ch

Abstract - For MinRes and SymmLQ it is essential to compute a QR decomposition of a tridiagonal coefficient matrix gained in the Lanczos process. This QR decomposition is constructed by an update scheme applying in every step a single Givens rotation. Using complex Householder reflections we generalize this idea to block tridiagonal matrices that occur in generalizations of MinRes and SymmLQ to block methods for systems with multiple right-hand sides. Some implementation details are given, and we compare the method with an algorithm based on Givens rotations used in block QMR. Our approach generalizes to the QR decomposition of upper block Hessenberg matrices resulting from the block Arnoldi process and is applicable in block GMRes.

Keywords— **block Lanczos process, block Krylov space methods.**

I. The symmetric Lanczos algorithm

In 1975 Christopher Paige and Michael Saunders [PAI 75] proposed two iterative Krylov subspace methods called MinRes and SymmLQ for solving sparse Hermitian indefinite linear systems

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{1}$$

with $\mathbf{A} = \mathbf{A}^{\mathsf{H}} \in \mathbb{C}^{N\times N}$, $\mathbf{x} \in \mathbb{C}^N$ and $\mathbf{b} \in \mathbb{C}^N$. An orthonormal basis for the Krylov subspace

$$\begin{aligned}\mathcal{K}_n(\mathbf{A},\mathbf{r}_0) &:= \mathsf{span}\left\{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \ldots, \mathbf{A}^{n-1}\mathbf{r}_0\right\} \\ &= \mathsf{span}\left\{\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{n-1}\right\}\end{aligned}$$

is constructed by the Lanczos process.
The smallest index n with $n = \dim \mathcal{K}_n(\mathbf{A},\mathbf{r}_0) = \dim \mathcal{K}_{n+1}(\mathbf{A},\mathbf{r}_0)$ is called the **grade of A with respect to $\mathbf{r}_0$** and denoted by $\bar{\nu}(\mathbf{A},\mathbf{r}_0)$.

Algorithm 1 (Sym. Lanczos algorithm).

Let a Hermitian matrix $\mathbf{A}$ *and a vector* $\mathbf{y}_0$ *with* $\|\mathbf{y}_0\|_2 = 1$ *be given. For constructing a nested set of orthonormal bases* $\{\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{m-1}\}$ *for the nested Krylov subspaces* $\mathcal{K}_m(\mathbf{A},\mathbf{y}_0)$ $(m = 1, 2, \ldots, \bar{\nu}(\mathbf{y}_0, \mathbf{A}))$ *compute, for* $n = 1, 2, \ldots,$ *the following:*

1) *Apply* $\mathbf{A}$ *to* $\mathbf{y}_{n-1} \perp \mathcal{K}_{n-1}(\mathbf{A},\mathbf{y}_0)$*:*

$$\widetilde{\mathbf{y}}_n := \mathbf{A}\mathbf{y}_{n-1}. \tag{2}$$

2) *Subtract the projections of* $\widetilde{\mathbf{y}}_n$ *on the last two basis vectors:*

$$\begin{aligned}\widetilde{\mathbf{y}}_n &:= \widetilde{\mathbf{y}}_n - \mathbf{y}_{n-2}\beta_{n-2} \quad \textit{if } n > 1, &(3)\\ \alpha_{n-1} &:= \langle \mathbf{y}_{n-1}, \widetilde{\mathbf{y}}_n\rangle, &(4)\\ \widetilde{\mathbf{y}}_n &:= \widetilde{\mathbf{y}}_n - \mathbf{y}_{n-1}\alpha_{n-1}. &(5)\end{aligned}$$

3) *Normalize the vector* $\widetilde{\mathbf{y}}_n \perp \mathcal{K}_n(\mathbf{A},\mathbf{y}_0)$ *if* $\|\widetilde{\mathbf{y}}_n\| > 0$*:*

$$\begin{aligned}\beta_{n-1} &:= \|\widetilde{\mathbf{y}}_n\|, &(6)\\ \mathbf{y}_n &:= \widetilde{\mathbf{y}}_n/\beta_{n-1}. &(7)\end{aligned}$$

The method stops when $\|\widetilde{\mathbf{y}}_n\| = 0$*. Then* $n = \bar{\nu}(\mathbf{A},\mathbf{y}_0)$.

For a basis of $\mathcal{K}_n(\mathbf{A},\mathbf{r}_0)$ we start with $\mathbf{y}_0 = \mathbf{r}_0/\|\mathbf{r}_0\|_2$. The coefficients of the recurrence relation in the Lanczos process are the entries of a tridiagonal matrix.

Theorem 1. *The vectors* $\left\{\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{\bar{\nu}(\mathbf{A},\mathbf{y}_0)-1}\right\}$ *constructed by this algorithm are orthonormal. For every* $n = 1, 2, \ldots \bar{\nu}(\mathbf{A},\mathbf{y}_0)$ *the first* n *vectors are a basis for* $\mathcal{K}_n(\mathbf{A},\mathbf{y}_0)$ *Moreover, for* $n < \bar{\nu}(\mathbf{A},\mathbf{y}_0)$

$$\mathbf{A}\mathbf{Y}_n = \mathbf{Y}_{n+1}\underline{\mathbf{T}}_n, \tag{8}$$

where

$$\mathbf{Y}_n = \begin{pmatrix} \mathbf{y}_0 & \mathbf{y}_1 & \ldots & \mathbf{y}_{n-1}\end{pmatrix},$$

$$\underline{\mathbf{T}}_n = \begin{pmatrix} \alpha_0 & \beta_0 & & & \\ \beta_0 & \alpha_1 & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \beta_{n-2} \\ & & & \beta_{n-2} & \alpha_{n-1} \\ \hline & & & & \beta_{n-1}\end{pmatrix} \in \mathbb{R}^{n+1\times n} \tag{9}$$

and $\beta_i > 0$ for all $i = 1, 2, \dots n-1$.

In MinRes and SymmLQ a QR decomposition of $\underline{\mathbf{T}}_n$ is used to construct approximations for the solution of the linear system (1). However, it is enough to apply an update scheme exploiting the tridiagonal structure of $\underline{\mathbf{T}}_n$. In every iteration only one Givens rotation has to be constructed.

II. A block Lanczos process

The Lanczos process can be generalized for multiple initial vectors, both in the Hermitian [CUL 74], [UND 75], [GOL 77], [LEW 77], [O'L 87] and the non-Hermitian case [O'L 87], [ALI 00], [BAI 99], [FRE 97b]. This idea is of interest when a linear system with s right-hand sides has to be solved or when eigenvalues of geometric multiplicity at most s have to be computed – an application not treated here. Approximating all systems in the same large space may accelerate convergence a lot. Usually that space is a direct sum of the corresponding Krylov spaces:

$$\begin{aligned}\mathcal{B}_n(\mathbf{A}, \mathbf{r}_0) &:= \bigoplus_{i=1}^{\mathsf{s}} \mathcal{K}_n\left(\mathbf{A}, \mathbf{r}_0^{(i)}\right) \\ &= \text{block span}\left\{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \dots, \mathbf{A}^{n-1}\mathbf{r}_0\right\}.\end{aligned}$$

The block Lanczos process creates in exact arithmetic the block vectors (which is just a fancy word for a "high and skinny" matrix to emphasize its character as a list of a few columns) $\mathbf{y}_0, \mathbf{y}_1, \dots, \mathbf{y}_{n-1}$. Their orthonormal columns are a basis for $\mathcal{B}_n(\mathbf{A}, \mathbf{v}_0)$.

$$\mathcal{B}_n(\mathbf{A}, \mathbf{r}_0) = \text{block span}\{\mathbf{y}_0, \mathbf{y}_1, \dots, \mathbf{y}_{n-1}\}.$$

Deflation is crucial in the underlying Lanczos process but not investigated in this paper. Deflation means to delete those columns of $\mathbf{y}_n$ which are already contained in $\mathcal{B}_n(\mathbf{A}, \mathbf{r}_0)$. The block vector $\mathbf{y}_i$ has s_i columns, where $\mathsf{s}_0 \geq \mathsf{s}_i \geq \mathsf{s}_{i+1}$, $i = 1, 2, \dots$
We denote by $\mathsf{t}_n(\mathbf{A}, \mathbf{v}_0)$ the dimension of $\mathcal{B}_n(\mathbf{A}, \mathbf{v}_0)$, which implies

$$\mathsf{t}_n(\mathbf{A}, \mathbf{v}_0) = \sum_{i=0}^{n-1} \mathsf{s}_i.$$

Definition 2. *The smallest index n with*

$$\mathsf{t}_n(\mathbf{A}, \mathbf{v}_0) = \mathsf{t}_{n+1}(\mathbf{A}, \mathbf{v}_0)$$

is called the **block grade of A with respect to $\mathbf{v}_0$** *and denoted by $\bar{\nu}(\mathbf{A}, \mathbf{v}_0)$.*

Algorithm 2 (Hermitian Block Lanczos algorithm).

Let a Hermitian matrix $\mathbf{A}$ and an orthonormal block vector $\mathbf{y}_0$ be given. For constructing a nested set of orthonormal block bases $\{\mathbf{y}_0, \mathbf{y}_1, \dots, \mathbf{y}_{m-1}\}$ for the nested Krylov subspaces $\mathcal{B}_m(\mathbf{A}, \mathbf{y}_0)$ $(m = 1, 2, \dots \leq \bar{\nu}(\mathbf{A}, \mathbf{y}_0))$ compute, for $n = 1, 2, \dots,$ the following:

1) *Apply $\mathbf{A}$ to $\mathbf{y}_{n-1} \perp \mathcal{B}_{n-1}(\mathbf{A}, \mathbf{y}_0)$:*

$$\widetilde{\mathbf{y}}_n := \mathbf{A}\mathbf{y}_{n-1}. \tag{10}$$

2) *Subtract the projections of $\widetilde{\mathbf{y}}_n$ on the last two basis block vectors:*

$$\widetilde{\mathbf{y}}_n := \widetilde{\mathbf{y}}_n - \mathbf{y}_{n-2}\boldsymbol{\beta}_{n-2}^{\mathsf{H}} \quad \text{if } n > 1, \tag{11}$$

$$\boldsymbol{\alpha}_{n-1} := \mathbf{y}_{n-1}^{\mathsf{H}}\widetilde{\mathbf{y}}_n, \tag{12}$$

$$\widetilde{\mathbf{y}}_n := \widetilde{\mathbf{y}}_n - \mathbf{y}_{n-1}\boldsymbol{\alpha}_{n-1}. \tag{13}$$

3) *QR factorization of $\widetilde{\mathbf{y}}_n \perp \mathcal{B}_n^{\square}(\mathbf{A}, \mathbf{y}_0)$ with $\operatorname{rank}\widetilde{\mathbf{y}}_n = \mathsf{s}_n \leq \mathsf{s}_{n-1}$:*

$$\begin{aligned}\widetilde{\mathbf{y}}_n &=: \begin{pmatrix} \mathbf{y}_n & \mathbf{y}_n^{\triangle} \end{pmatrix} \begin{pmatrix} \boldsymbol{\rho}_n & \boldsymbol{\rho}_n^{\square} \\ \mathbf{0} & \boldsymbol{\rho}_n^{\triangle} \end{pmatrix} \boldsymbol{\pi}_n^{\mathsf{T}} \\ &=: \begin{pmatrix} \mathbf{y}_n & \mathbf{y}_n^{\triangle} \end{pmatrix} \begin{pmatrix} \boldsymbol{\beta}_{n-1} \\ \boldsymbol{\beta}_{n-1}^{\triangle} \end{pmatrix},\end{aligned} \tag{14}$$

where:

$\boldsymbol{\pi}_n$	*is an $\mathsf{s}_{n-1} \times \mathsf{s}_{n-1}$ permutation matrix.*
$\mathbf{y}_n$	*is an $N \times \mathsf{s}_n$ block vector with full numerical column rank, which goes into the basis.*
$\mathbf{y}_n^{\triangle}$	*is an $N \times (\mathsf{s}_{n-1} - \mathsf{s}_n)$ matrix that will be deflated,*
$\boldsymbol{\rho}_n$	*is an $\mathsf{s}_n \times \mathsf{s}_n$ upper triangular, nonsingular matrix.*
$\boldsymbol{\rho}_n^{\square}$	*is an $\mathsf{s}_n \times (\mathsf{s}_{n-1} - \mathsf{s}_n)$ matrix.*
$\boldsymbol{\rho}_n^{\triangle}$	*is an upper triangular $(\mathsf{s}_{n-1} - \mathsf{s}_n) \times (\mathsf{s}_{n-1} - \mathsf{s}_n)$ matrix with $\lVert\boldsymbol{\rho}_n^{\triangle}\rVert_F = O(\sigma_{\mathsf{s}_n+1})$, where σ_{s_n+1} is the largest singular value of $\widetilde{\mathbf{y}}_n$ smaller or equal to* tol.

The permutations are encapsulated in the block coefficients $\boldsymbol{\beta}_i$. Let

$$\mathbf{P}_n :\equiv \text{block diag}(\boldsymbol{\pi}_1, \dots, \boldsymbol{\pi}_n)$$

be the permutation matrix that describes all these permutations. Note that $\mathbf{P}_n^{\mathsf{T}} = \mathbf{P}_n^{-1}$. If tol $= 0$ we speak of **exact deflation**. The algorithm uses block permutations as well as deflation. It is therefore not immediately apparent that the fundamental Lanczos relationships still hold.

THEOREM 3. *With exact deflation the columns of the block vectors* $\mathbf{y}_0, \mathbf{y}_1, \ldots$ *constructed by Algorithm 2 are orthonormal. For every* $n = 1, 2, \ldots, \bar{\nu}^{\square}(\mathbf{A}, \mathbf{y}_0)$ *the columns of the first n block vectors are an orthonormal basis for* $\mathcal{B}_n(\mathbf{A}, \mathbf{y}_0)$. *Moreover, for* $n < \bar{\nu}^{\square}(\mathbf{A}, \mathbf{y}_0)$

$$\mathbf{A}\mathbf{Y}_n = \mathbf{Y}_{n+1}\underline{\mathbf{T}}_n, \tag{15}$$

where

$$\mathbf{Y}_n = \begin{pmatrix} \mathbf{y}_0 & \mathbf{y}_1 & \ldots & \mathbf{y}_{n-1} \end{pmatrix},$$

$$\underline{\mathbf{T}}_n = \left(\begin{array}{cccccc} \boldsymbol{\alpha}_0 & \boldsymbol{\beta}_0^{\mathsf{H}} & & & & \\ \boldsymbol{\beta}_0 & \boldsymbol{\alpha}_1 & \ddots & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \boldsymbol{\beta}_{n-2}^{\mathsf{H}} \\ & & & \boldsymbol{\beta}_{n-2} & \boldsymbol{\alpha}_{n-1} \\ \hline & & & & \boldsymbol{\beta}_{n-1} \end{array}\right) \in \mathbb{C}^{\mathsf{t}_{n+1}\times\mathsf{t}_n} \tag{16}$$

and $\boldsymbol{\alpha}_i = \boldsymbol{\alpha}_i^{\mathsf{H}}$ *for all* $i = 1, 2, \ldots n-1$.

III. An update scheme for the QR decomposition of block tridiagonal matrices

The subdiagonal blocks of $\underline{\mathbf{T}}_n\mathbf{P}_n$ are upper trapezoidal. This is used to construct an update scheme for this block tridiagonal matrix.

Let $\underline{\mathbf{T}}_n\mathbf{P}_n = \mathbf{Q}_{n+1}\underline{\mathbf{R}}_n$ be the QR decomposition of $\underline{\mathbf{T}}_n\mathbf{P}_n$ so that $\mathbf{Q}_{n+1}$ is a unitary matrix and $\underline{\mathbf{R}}_n$ is an upper triangular matrix with full column rank. The block tridiagonal matrix $\underline{\mathbf{T}}_n\mathbf{P}_n$ is

$$\underline{\mathbf{T}}_n\mathbf{P}_n = \left(\begin{array}{ccccc} \boldsymbol{\alpha}_0\boldsymbol{\pi}_1 & \boldsymbol{\beta}_0^{\mathsf{H}}\boldsymbol{\pi}_2 & & & \\ \boldsymbol{\beta}_0\boldsymbol{\pi}_1 & \boldsymbol{\alpha}_1\boldsymbol{\pi}_2 & \ddots & & \\ & \boldsymbol{\beta}_1\boldsymbol{\pi}_2 & \ddots & \ddots & \\ & & \ddots & \ddots & \boldsymbol{\beta}_{n-2}^{\mathsf{H}}\boldsymbol{\pi}_n \\ & & & \ddots & \boldsymbol{\alpha}_{n-1}\boldsymbol{\pi}_n \\ \hline & & & & \boldsymbol{\beta}_{n-1}\boldsymbol{\pi}_n \end{array}\right) \tag{17}$$

where:

$\boldsymbol{\alpha}_i\boldsymbol{\pi}_{i+1}$	is an $\mathsf{s}_i \times \mathsf{s}_i$ block and
$\boldsymbol{\beta}_i\boldsymbol{\pi}_{i+1}$	is an $\mathsf{s}_{i+1} \times \mathsf{s}_i$ upper trapezoidal block, $\boldsymbol{\beta}_i\boldsymbol{\pi}_{i+1} = \begin{pmatrix} \boldsymbol{\rho}_{i+1} & \boldsymbol{\rho}_{i+1}^{\square} \end{pmatrix}$

The $\mathsf{t}_{n+1} \times \mathsf{t}_n$ matrix $\underline{\mathbf{R}}_n$ has the form

$$\underline{\mathbf{R}}_n = \left(\begin{array}{ccccc} \widetilde{\boldsymbol{\alpha}}_0 & \widetilde{\boldsymbol{\beta}}_0 & \widetilde{\boldsymbol{\gamma}}_0 & & \\ & \widetilde{\boldsymbol{\alpha}}_1 & \widetilde{\boldsymbol{\beta}}_1 & \ddots & \\ & & \ddots & \ddots & \widetilde{\boldsymbol{\gamma}}_{n-3} \\ & & & \ddots & \widetilde{\boldsymbol{\beta}}_{n-2} \\ & & & & \widetilde{\boldsymbol{\alpha}}_{n-1} \\ \hline & & \mathbf{0}_{\mathsf{s}_n\times\mathsf{t}_n} & & \end{array}\right) \tag{18}$$

where:

$\widetilde{\boldsymbol{\alpha}}_i$	is an $\mathsf{s}_i \times \mathsf{s}_i$ upper triangular block with full rank,
$\widetilde{\boldsymbol{\beta}}_i$	is an $\mathsf{s}_i \times \mathsf{s}_{i+1}$ block and
$\widetilde{\boldsymbol{\gamma}}_i$	is an $\mathsf{s}_i \times \mathsf{s}_{i+2}$ lower trapezoidal block.

We determine the unitary matrix $\mathbf{Q}_{n+1}$ in its factored form. With

$$\mathbf{Q}_1 = \mathbf{I}_{\mathsf{s}_0} \tag{19}$$

we introduce the recurrence relation

$$\mathbf{Q}_{n+1} := \left(\begin{array}{c|c} \mathbf{Q}_n & \mathbf{0}_{\mathsf{t}_n\times\mathsf{s}_n} \\ \hline \mathbf{0}_{\mathsf{s}_n\times\mathsf{t}_n} & \mathbf{I}_{\mathsf{s}_n} \end{array}\right)\mathbf{U}_n \tag{20}$$

where

$$\mathbf{U}_n := \left(\begin{array}{c|c} \mathbf{I}_{\mathsf{t}_{n-1}} & \mathbf{0}_{\mathsf{t}_{n-1}\times(\mathsf{s}_{n-1}+\mathsf{s}_n)} \\ \hline \mathbf{0}_{(\mathsf{s}_{n-1}+\mathsf{s}_n)\times\mathsf{t}_{n-1}} & \hat{\mathbf{U}}_n \end{array}\right) \tag{21}$$

Here $\hat{\mathbf{U}}_n$ is a unitary $(\mathsf{s}_{n-1}+\mathsf{s}_n) \times (\mathsf{s}_{n-1}+\mathsf{s}_n)$ matrix. Given a sequence of unitary transformations $\hat{\mathbf{U}}_1, \ldots \hat{\mathbf{U}}_n$ it is possible to compute $\mathbf{Q}_{n+1}$ with a simple scheme. If the matrix $\mathbf{Q}_n$ has the form

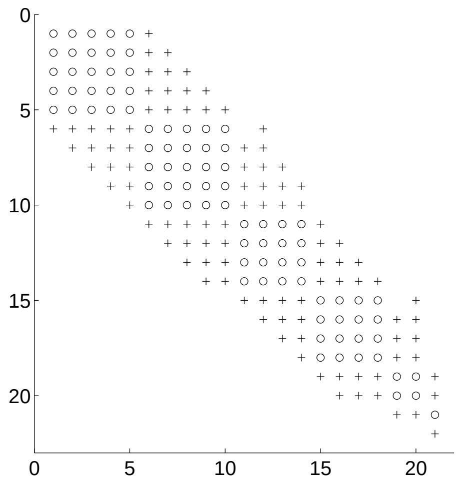
$$\mathbf{Q}_n = \begin{pmatrix} \mathbf{q}_0 & \mathbf{q}_1 & \ldots & \mathbf{q}_{n-2} & \widetilde{\mathbf{q}}_{n-1} \end{pmatrix}$$

where $\mathbf{q}_i \in \mathbb{C}^{\mathsf{t}_n\times\mathsf{s}_i}$ and

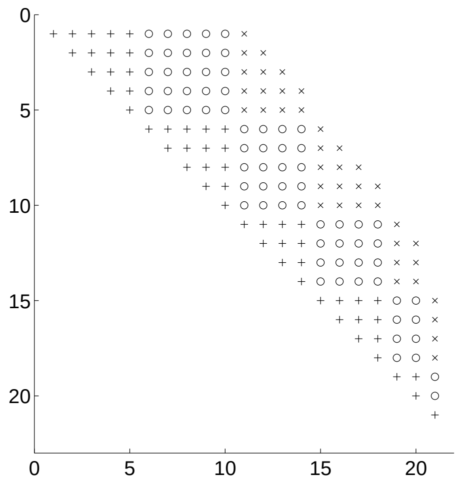
$$\hat{\mathbf{U}}_n = \begin{pmatrix} \hat{\mathbf{U}}_{n,u} \\ \hat{\mathbf{U}}_{n,d} \end{pmatrix} = \begin{pmatrix} \hat{\mathbf{U}}_{n,u,l} & \hat{\mathbf{U}}_{n,u,r} \\ \hat{\mathbf{U}}_{n,d,l} & \hat{\mathbf{U}}_{n,d,r} \end{pmatrix} \tag{22}$$

where:

$\hat{\mathbf{U}}_{n,u}$	is an $\mathsf{s}_{i-1} \times \mathsf{s}_{i-1} + \mathsf{s}_i$ matrix,
$\hat{\mathbf{U}}_{n,d}$	is an $\mathsf{s}_i \times \mathsf{s}_{i-1} + \mathsf{s}_i$ matrix,
$\hat{\mathbf{U}}_{n,u,l}$	is an $\mathsf{s}_{i-1} \times \mathsf{s}_{i-1}$ matrix,
$\hat{\mathbf{U}}_{n,u,r}$	is an $\mathsf{s}_{i-1} \times \mathsf{s}_i$ matrix,
$\hat{\mathbf{U}}_{n,d,l}$	is an $\mathsf{s}_i \times \mathsf{s}_{i-1}$ matrix,
$\hat{\mathbf{U}}_{n,d,r}$	is an $\mathsf{s}_i \times \mathsf{s}_i$ matrix,



(a) *The block tridiagonal structure of a matrix* $\underline{\mathbf{T}}_n\mathbf{P}_n$*. The matrix is not symmetric, but all subdiagonal blocks are upper trapezoidal.*



(b) *The corresponding block structure of the matrix* $\underline{\mathbf{R}}_n$*.*

then

$$\mathbf{Q}_{n+1} = \begin{pmatrix} \cdots & \mathbf{q}_{n-2} & \widetilde{\mathbf{q}}_{n-1}\hat{\mathbf{U}}_{n,u,l} & \widetilde{\mathbf{q}}_{n-1}\hat{\mathbf{U}}_{n,u,r} \\ & & \hat{\mathbf{U}}_{n,d,l} & \hat{\mathbf{U}}_{n,d,r} \end{pmatrix} \tag{23}$$

In particular the matrix $\mathbf{Q}_{n+1}$ has a trapezoidal structure. The update scheme (23) yields the following algorithm, which we describe using MATLAB notation:

ALGORITHM 3 (CONSTRUCTION OF $\mathbf{Q}_{n+1}$).

Let $\hat{\mathbf{U}}_1, \dots \hat{\mathbf{U}}_n$ *be the constructed sequence of unitary matrices where* $\hat{\mathbf{U}}_i$ *is of order* $\mathsf{s}_{i-1}+\mathsf{s}_i$ *and let* $\mathbf{Q}_{n+1} = \mathbf{I}_{\mathsf{t}_{n+1}}$*. For the recursive definition of* $\mathbf{Q}_{n+1}$*, for* $i = 1, \dots, n$*:*

- *Apply the upper part of* $\hat{\mathbf{U}}_i$*:*

$$\mathbf{Q}_{n+1}\left(1:\mathsf{t}_i, \mathsf{t}_{i-1}+1:\mathsf{t}_{i+1}\right) = \mathbf{Q}_{n+1}\left(1:\mathsf{t}_i, \mathsf{t}_{i-1}+1:\mathsf{t}_i\right)\hat{\mathbf{U}}_{i,u}$$

- *Apply the lower part of* $\hat{\mathbf{U}}_i$*:*

$$\mathbf{Q}_{n+1}\left(\mathsf{t}_i+1:\mathsf{t}_{i+1}, \mathsf{t}_{i-1}+1:\mathsf{t}_{i+1}\right) = \hat{\mathbf{U}}_{i,d}. \tag{24}$$

The multiplication of $\underline{\mathbf{T}}_n\mathbf{P}_n$ by $\mathbf{Q}_n^{\mathsf{H}}$ annihilates all subdiagonal elements except those below the diagonal of the last s_{n-1} columns, or if we regard $\underline{\mathbf{T}}_n\mathbf{P}_n$ as a matrix with n block columns, it does not annihilate the subdiagonal elements in the last block column, i.e.

$$\begin{pmatrix} \mathbf{Q}_n^{\mathsf{H}} & \\ & \mathbf{I}_{\mathsf{s}_n} \end{pmatrix} \underline{\mathbf{T}}_n\mathbf{P}_n = \begin{pmatrix} \widetilde{\boldsymbol{\alpha}}_0 & \widetilde{\boldsymbol{\beta}}_0 & \widetilde{\boldsymbol{\gamma}}_0 & & & \\ & \widetilde{\boldsymbol{\alpha}}_1 & \widetilde{\boldsymbol{\beta}}_1 & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & \ddots & \widetilde{\boldsymbol{\gamma}}_{n-3} \\ & & & & \widetilde{\boldsymbol{\alpha}}_{n-2} & \widetilde{\boldsymbol{\beta}}_{n-2} \\ & & & & & \boldsymbol{\mu}_n \\ \hline & & & & & \boldsymbol{\beta}_{n-1}\boldsymbol{\pi}_n \end{pmatrix}. \tag{25}$$

For the entries in the last block column, respectively the entries in the last s_{n-1} columns we have

$$\begin{pmatrix} \widetilde{\boldsymbol{\gamma}}_{n-3} \\ \widetilde{\boldsymbol{\beta}}_{n-2} \\ \boldsymbol{\mu}_n \end{pmatrix} = \begin{pmatrix} \mathbf{I}_{\mathsf{s}_{n-3}} & \\ & \hat{\mathbf{U}}_{n-1}^{\mathsf{H}} \end{pmatrix} \begin{pmatrix} \hat{\mathbf{U}}_{n-2}^{\mathsf{H}} & \\ & \mathbf{I}_{\mathsf{s}_{n-1}} \end{pmatrix} \begin{pmatrix} \mathbf{0}_{\mathsf{s}_{n-3}\times\mathsf{s}_{n-1}} \\ \boldsymbol{\beta}_{n-2}^{\mathsf{H}}\boldsymbol{\pi}_n \\ \boldsymbol{\alpha}_{n-1}\boldsymbol{\pi}_n \end{pmatrix}. \tag{26}$$

For annihilating all subdiagonal elements we have to construct a unitary matrix $\mathbf{U}_n'$ such that

$$\begin{pmatrix} \boldsymbol{\mu}_n \\ \boldsymbol{\beta}_{n-1}\boldsymbol{\pi}_n \end{pmatrix} = \hat{\mathbf{U}}_n \begin{pmatrix} \widetilde{\boldsymbol{\alpha}}_{n-1} \\ \mathbf{0}_{\mathsf{s}_n\times\mathsf{s}_{n-1}} \end{pmatrix}, \tag{27}$$

where $\widetilde{\boldsymbol{\alpha}}_{n-1}$ is an upper triangular nonsingular block vector.

Algorithm 4 (An update scheme for the QR decomposition).

By applying a sequence of unitary matrices $\hat{\mathbf{U}}_1, \ldots, \hat{\mathbf{U}}_m$ *on the matrix* $\underline{\mathbf{T}}_m \mathbf{P}_m$ *of (17) the upper triangular matrix* $\mathbf{R}_m$ *of (18) is recursively constructed as follows. For* $n = 1, \ldots, m$*:*

1) *Let* $\widetilde{\boldsymbol{\alpha}}_{n-1} := \boldsymbol{\alpha}_{n-1}\boldsymbol{\pi}_n$*, and* $\widetilde{\boldsymbol{\beta}}_{n-2} := \boldsymbol{\beta}^{\mathsf{H}}_{n-2}\boldsymbol{\pi}_n$ *if* $n > 1$.
If $n > 2$*, apply* $\mathbf{U}^{\mathsf{H}}_{n-2}$ *to the new last column of* $\underline{\mathbf{T}}_n$*:*

$$\begin{pmatrix} \widetilde{\boldsymbol{\gamma}}_{n-3} \\ \widetilde{\boldsymbol{\beta}}_{n-2} \end{pmatrix} := \hat{\mathbf{U}}^{\mathsf{H}}_{n-2} \begin{pmatrix} \mathbf{0}_{\mathsf{s}_{n-3}\times\mathsf{s}_{n-1}} \\ \widetilde{\boldsymbol{\beta}}_{n-2} \end{pmatrix};$$

if $n > 1$*, apply* $\mathbf{U}^{\mathsf{H}}_{n-1}$ *to the last column of* $\mathbf{U}^{\mathsf{H}}_{n-2}\underline{\mathbf{T}}_n$*:*

$$\begin{pmatrix} \widetilde{\boldsymbol{\beta}}_{n-2} \\ \widetilde{\boldsymbol{\alpha}}_{n-1} \end{pmatrix} := \hat{\mathbf{U}}^{\mathsf{H}}_{n-1} \begin{pmatrix} \widetilde{\boldsymbol{\beta}}_{n-2} \\ \widetilde{\boldsymbol{\alpha}}_{n-1} \end{pmatrix}.$$

2) *Let* $\boldsymbol{\mu}_n := \widetilde{\boldsymbol{\alpha}}_{n-1}$ *and compute* $\hat{\mathbf{U}}^{\mathsf{H}}_n$ *according to (27).*
3) *Compute* $\mathbf{Q}_{n+1}$ *according to Algorithm 3.*

There are various possible ways to construct such a matrix $\hat{\mathbf{U}}_n$. Freund and Malhotra [FRE 97a] apply Givens rotations since a problem with a single right-hand side should be a special case of the general block problem. In the case of one right-hand side it is enough to apply a single Givens rotation. So the rationale behind this approach was to generalize this special case [FRE 04]. For our goals the most efficient way is to use a product of complex Householder reflections.

IV. Complex Householder reflections

Let $\mathbf{y} = \begin{pmatrix} y_1 & \ldots y_n \end{pmatrix}^{\mathsf{T}}$ be a complex vector. The goal is to construct a unitary matrix $\mathbf{U} \in \mathbb{C}^{n\times n}$ such that

$$\mathbf{U}\mathbf{y} = \alpha\mathbf{e}_1$$

where $\alpha \in \mathbb{C}$. As $\mathbf{U}$ is unitary we note that $|\alpha| = \|\mathbf{y}\|_2$. for any nonzero $\mathbf{v} \in \mathbb{C}^n$ the matrix

$$\mathbf{H}_{\mathbf{v}} = \mathbf{I}_n - 2\frac{\mathbf{v}\,\mathbf{v}^{\mathsf{H}}}{\langle \mathbf{v}, \mathbf{v}\rangle} = \mathbf{I}_n + \beta\,\mathbf{v}\,\mathbf{v}^{\mathsf{H}}, \qquad (28)$$

where $\beta = -2/\langle \mathbf{v}, \mathbf{v}\rangle \in \mathbb{R}$, is called a Householder reflection. The matrix $\mathbf{H}_{\mathbf{v}}$ describes a reflection at the complimentary subspace orthogonal to $\mathbf{v}$. We note that $\mathbf{H}_{\mathbf{v}}$ is Hermitian and unitary, i.e. $\mathbf{H}^{\mathsf{H}}_{\mathbf{v}} = \mathbf{H}_{\mathbf{v}}$ and $\mathbf{H}_{\mathbf{v}}\mathbf{H}^{\mathsf{H}}_{\mathbf{v}} = \mathbf{I}_n$. The vector $\mathbf{y}$ is mapped to

$$\mathbf{H}_{\mathbf{v}}\mathbf{y} = \mathbf{y} + \beta\mathbf{v}\,\langle \mathbf{v}, \mathbf{y}\rangle\,. \qquad (29)$$

Keeping in mind our goal we demand

$$\mathbf{H}_{\mathbf{v}}\mathbf{y} = \alpha\mathbf{e}_1.$$

This yields

$$\mathbf{y} - \alpha\mathbf{e}_1 = -\beta\,\langle \mathbf{v}, \mathbf{y}\rangle\,\mathbf{v}. \qquad (30)$$

In particular $\mathbf{v} \in \mathsf{span}\,\{\mathbf{y} - \alpha\mathbf{e}_1\}$. As $\mathbf{H}_{\mathbf{v}} = \mathbf{H}_{\lambda\mathbf{v}}$ for all $\lambda \in \mathbb{C} - \{0\}$ we can choose $\mathbf{v} = \mathbf{y} - \alpha\mathbf{e}_1$ without loss of generality. By (30) this choice implies

$$\langle \mathbf{v}, \mathbf{y}\rangle = -\beta^{-1} \in \mathbb{R}.$$

Let $y_1 = |y_1|e^{i\theta}$ and $\alpha = \|\mathbf{y}\|_2 e^{i\theta_\alpha}$. Then

$$\langle \mathbf{v}, \mathbf{y}\rangle = \langle \mathbf{y} - \alpha\mathbf{e}_1, \mathbf{y}\rangle = \|\mathbf{y}\|_2^2 - \|\mathbf{y}\|_2 e^{-i\theta_\alpha}e^{i\theta}|y_1|.$$

So either $\alpha = +\|\mathbf{y}\|_2 e^{i\theta}$ or $\alpha = -\|\mathbf{y}\|_2 e^{i\theta}$. The second choice is better, otherwise cancellation effects may occur in the first component of $\mathbf{v}$. Finally we note that

$$-\beta^{-1} = \langle \mathbf{v}, \mathbf{y}\rangle = \|\mathbf{y}\|_2\left(\|\mathbf{y}\|_2 + |y_1|\right).$$

Algorithm 5 (Implicit construction of $\mathbf{H}_{\mathbf{v}}$).

Let $\mathbf{y} = \begin{pmatrix} y_1 & \ldots & y_n \end{pmatrix}^{\mathsf{T}}$ *be a complex vector. A Householder reflection* $\mathbf{H}_{\mathbf{v}}$ *of order* n *with* $\mathbf{H}_{\mathbf{v}}\mathbf{y} = \alpha\mathbf{e}_1$ *is constructed as follows. Let* $y_1 = |y_1|e^{i\theta}$.

- *Compute* α *and* β*:*

$$\alpha = -\|\mathbf{y}\|_2 e^{i\theta}, \qquad \beta = \frac{-1}{\|\mathbf{y}\|_2\left(\|\mathbf{y}\|_2 + |y_1|\right)}. \qquad (31)$$

- *Compute the vector* $\mathbf{v}$*:*

$$\mathbf{v} = \mathbf{y} - \alpha\mathbf{e}_1. \qquad (32)$$

It is not necessary to compute the actual matrix $\mathbf{H}_{\mathbf{v}}$. It is much more economical and accurate to store the vector $\mathbf{v}$ and the coefficient β and apply the identity (29). Parlett [PAR 71] presents a thorough discussion on the choice of the sign of α when computing Householder reflectors. Lehoucq [LEH 96] compares different variants for the choice of the vector $\mathbf{v}$ and the corresponding coefficient β. He specifically compares the different computations of an elementary unitary matrix in EISPACK, LINPACK, NAG and LAPACK. The above scheme is due to Wilkinson [WIL 65, pp. 49-50].

V. QR decomposition of a lower banded matrix

The idea is to use the trapezoidal structure of the matrix $\boldsymbol{\nu}_n := \boldsymbol{\beta}_{n-1}\boldsymbol{\pi}_n$ in (27). Recall that $\boldsymbol{\mu}_n$ is $\mathsf{s}_n \times \mathsf{s}_n$, while $\boldsymbol{\nu}_n$ is $\mathsf{s}_n \times \mathsf{s}_{n-1}$. For example, if $\mathsf{s}_{n-1} = 5$ and $\mathsf{s}_n = 4$,

$$\begin{pmatrix} \boldsymbol{\mu}_n \\ \boldsymbol{\nu}_n \end{pmatrix} = \left(\begin{array}{ccccc} \circ & \circ & \circ & \circ & \circ \\ \circ & \circ & \circ & \circ & \circ \\ \circ & \circ & \circ & \circ & \circ \\ \circ & \circ & \circ & \circ & \circ \\ \circ & \circ & \circ & \circ & \circ \\ \hline \circ & \circ & \circ & \circ & \circ \\ & \circ & \circ & \circ & \circ \\ & & \circ & \circ & \circ \\ & & & \circ & \circ \end{array}\right).$$

We determine s_{n-1} Householder reflections $\mathbf{H}_{1,n}, \ldots, \mathbf{H}_{\mathsf{s}_{n-1},n}$ such that

$$\begin{pmatrix} \widetilde{\boldsymbol{\alpha}}_{n-1} \\ \mathbf{0}_{\mathsf{s}_n \times \mathsf{s}_{n-1}} \end{pmatrix} = \mathbf{H}_{\mathsf{s}_{n-1},n} \ldots \mathbf{H}_{1,n} \begin{pmatrix} \boldsymbol{\mu}_n \\ \boldsymbol{\nu}_n \end{pmatrix}, \tag{33}$$

where $\widetilde{\boldsymbol{\alpha}}_{n-1}$ is an upper triangular matrix. In particular

$$\hat{\mathbf{U}}_n = \mathbf{H}_{1,n} \ldots \mathbf{H}_{\mathsf{s}_{n-1},n}. \tag{34}$$

Assume that reflections $\mathbf{H}_{1,n}, \mathbf{H}_{2,n}$ have been computed such that

$$\mathbf{H}_{2,n}\mathbf{H}_{1,n} \begin{pmatrix} \boldsymbol{\mu}_n \\ \boldsymbol{\nu}_n \end{pmatrix} = \left(\begin{array}{ccccc} \circ & \circ & \circ & \circ & \circ \\ & \circ & \circ & \circ & \circ \\ & & \bullet & \circ & \circ \\ & & \bullet & \circ & \circ \\ & & \bullet & \circ & \circ \\ \hline & & \bullet & \circ & \circ \\ & & \bullet & \circ & \circ \\ & & \bullet & \circ & \circ \\ & & & \circ & \circ \end{array}\right).$$

The highlighted section of the third column vector determines the next Householder reflection. In step i this vector has the size

$$l_{i,n} = \underbrace{\mathsf{s}_{n-1} - i + 1}_{\text{size of upper part}} + \underbrace{\min(i, \mathsf{s}_n)}_{\text{size of lower part}},$$

and the last entry is in row

$$e_{i,n} = l_{i,n} + i - 1.$$

In this example we have

$$l_{3,n} = 5 - 3 + 1 + 3 = 6, \qquad e_{i,n} = 6 + 3 - 1 = 8.$$

Hence the Householder reflection is given by

$$\mathbf{H}_{i,n} = \operatorname{diag}\left(\mathbf{I}_{i-1}, \hat{\mathbf{H}}_{i,n}, \mathbf{I}_{\mathsf{s}_n - \min(i,\mathsf{s}_n)}\right),$$

where $\hat{\mathbf{H}}_{i,n}$ is a Householder reflection in the sense of our original definition (28): a reflection at a hyperplane but in a space of dimension $l_{i,n}$ only. When we apply this reflection we only compute those entries which are not invariant. In this example the first two and the last row would be not influenced at all. All we have to do is to apply the reflection $\hat{\mathbf{H}}_{i,n}$ on the submatrix whose left column is exactly given by the vector generating $\hat{\mathbf{H}}_{i,n}$. Here the submatrix is highlighted:

$$\left(\begin{array}{ccccc} \circ & \circ & \circ & \circ & \circ \\ & \circ & \circ & \circ & \circ \\ & & \alpha & \bullet & \bullet \\ & & & \bullet & \bullet \\ & & & \bullet & \bullet \\ \hline & & & \bullet & \bullet \\ & & & \bullet & \bullet \\ & & & \bullet & \bullet \\ & & & \circ & \circ \end{array}\right)$$

This submatrix is updated and we proceed with the construction of the next reflection.

Algorithm 6 (Implicit construction of $\hat{\mathbf{U}}_n$).

Let $\boldsymbol{\mu}_n$ a $\mathsf{s}_{n-1} \times \mathsf{s}_{n-1}$ and $\boldsymbol{\nu}_n$ an upper trapezoidal $\mathsf{s}_n \times \mathsf{s}_{n-1}$ block. In a implicit way we construct s_{n-1} Householder reflections such that (33) holds. Let

$$\mathbf{M} = \begin{pmatrix} \boldsymbol{\mu}_n \\ \boldsymbol{\nu}_n \end{pmatrix}.$$

For $i = 1, \ldots, \mathsf{s}_{n-1}$:

- *Compute $l_{i,n}$ and $e_{i,n}$:*

$$l_{i,n} = \mathsf{s}_{n-1} - i + 1 + \min(i, \mathsf{s}_n), \quad e_{i,n} = l_{i,n} + i - 1. \tag{35}$$

- *Create implicitly by Algorithm 5 the Householder reflection $\hat{\mathbf{H}}_{i,n}$, that is compute β and the vector $\mathbf{v}$, using the vector*

$$\mathbf{y}_{i,n} = \mathbf{M}(i : e_{i,n}, i). \tag{36}$$

- *Apply $\hat{\mathbf{H}}_{i,n}$ to the corresponding submatrix of $\mathbf{M}$:*

$$\begin{aligned} \mathbf{M}(i : e_{i,n}, i+1 : \mathsf{s}_{n-1}) = {} & \mathbf{M}(i : e_{i,n}, i+1 : \mathsf{s}_{n-1}) \\ & + \beta \mathbf{v}\left(\mathbf{v}^{\mathsf{H}} \mathbf{M}(i : e_{i,n}, i+1 : \mathsf{s}_{n-1})\right). \end{aligned} \tag{37}$$

VI. Householder reflections vs. Givens rotations

Given an upper trapezoidal matrix $\mathbf{M}$ with $\mathsf{s}_1 = \mathsf{s}_2 = w$, Householder reflections, applied as described in Section IV, are an efficient way to construct the QR decomposition of $\mathbf{M}$. An alternative are Givens rotations: instead of the Householder reflection of Algorithm 5 a set of $w - 1$ Givens rotations is applied in

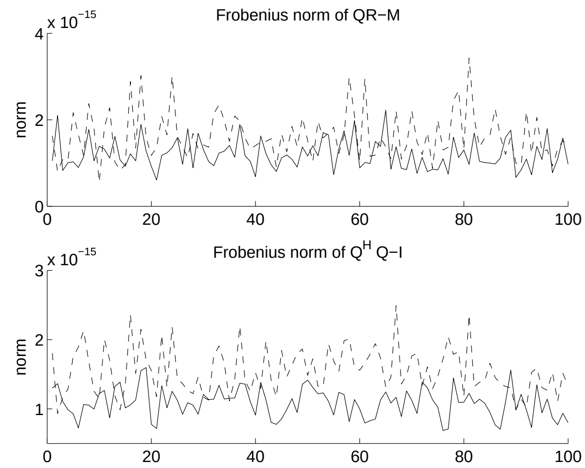


Fig. 1. Experiment 1: Accuracy of the QR decomposition of 100 random 10×5 upper trapezoidal matrices $\mathbf{M}$. The solid line represents results gained by using Householder reflections. The dashed line corresponds to Givens rotations.

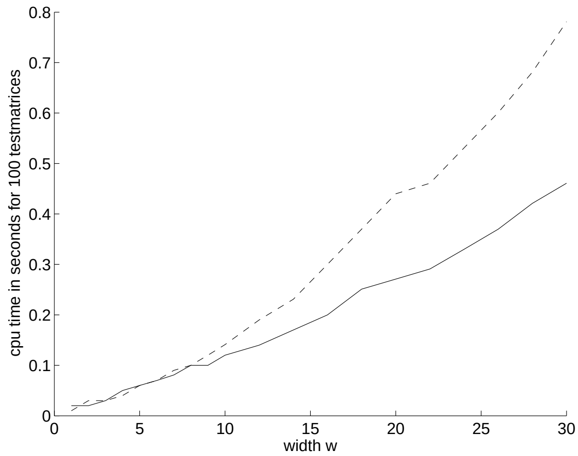


Fig. 2. Experiment 2: Computation time for the QR decomposition of 100 random $2w \times w$ upper trapezoidal matrices $\mathbf{M}$. The solid line represents results gained by using Householder reflections. The dashed line corresponds to Givens rotations. In both cases we have updated the remaining columns simultaneously.

an explicit loop. Therefore $w - 1$ roots have to be computed. In Algorithm 6 the block-wise update in (37) by a single Householder reflection (applied to a whole submatrix of $\mathbf{M}$) must be replaced by applying the $w - 1$ Givens rotations to suitable rows of this submatrix of $\mathbf{M}$. In a first experiment we compare the accuracy of both approaches.

Experiment 1. *We apply both approaches for the QR decomposition — Householder reflections and Givens rotations — to a set of* 100 *random* 10×5 *upper trapezoidal matrices* $\mathbf{M}$. *The results are shown in Fig. 1. The accuracy of both methods turns out to be on the same level: except in a few cases the Frobenius norms of* $\mathbf{M} - \mathbf{QR}$ *and of* $\mathbf{Q}^{\mathsf{H}}\mathbf{Q} - \mathbf{I}$ *are for both methods of the same magnitude. The norm of* $\mathbf{Q}^{\mathsf{H}}\mathbf{Q} - \mathbf{I}$ *is typically slightly smaller if Householder reflections are used. For the explicit computation of* $\mathbf{Q}$ *we apply the unitary transformations to* $\mathbf{I}_{10\times 10}$ *as in (34).*

Next we compare the computing time of the two approaches. The time-critical second step involves the application of the implicitly constructed unitary matrix to the remaining columns. The update in (37), where a Householder reflection is applied to all these columns at once, can be done by BLAS2 operations, whereas with Givens rotations we have to apply the corresponding BLAS1 operations in a loop from the bottom row to the top row of the remaining submatrix of $\mathbf{M}$.

Experiment 2. *In order to compare the speed of the two approaches we measure the cpu time for constructing the QR decomposition of* 100 *random matrices* M *of size* $\mathsf{s}_1 = \mathsf{s}_2 = w$, *using Householder reflections or Givens rotations, respectively. See Fig. 2. For a matrix* $\mathbf{M}$ *of width 30 the Givens rotations turn out to be about 70% slower, but for small matrices the efficiency difference is small.*

The block Lanczos process introduced in this paper allows a block-wise construction of $\underline{\mathbf{T}}_n$ as opposed to the column-wise construction of Ruhe [RUH 79] or Freund and Malhotra [FRE 97a]. The block-wise construction of $\underline{\mathbf{T}}_n$ allows us to update its QR decomposition block-wise. Hence we can profit from the possibility to update several columns at once in (37). (Recall that in this application $\mathbf{M}$ is a submatrix of $\underline{\mathbf{T}}_n$.) In contrast, in the implementation of block QMR by Freund and Malhotra there is only a routine for rotating a single vector by a set of Givens rotations, since in block QMR the matrix $\underline{\mathbf{T}}_n$ is generated column-wise, and its QR decomposition is updated whenever a new column of $\underline{\mathbf{T}}_n$ becomes available. In particular, in the first iteration a vector of length $\mathsf{s} + 1$ is obtained as first column of $\underline{\mathbf{T}}_n$. A set of s Givens rotations is constructed to annihilate all except the first entry. In the next iteration a second vector of length $\mathsf{s} + 2$ appears as second column, etc. At every stage we have to apply the previously constructed rotations or reflections. Hence, instead of applying a reflection at once on a matrix of width $k < \mathsf{s}$ it is necessary to apply a reflection or a set of s rotations k times on a column vector. This is equivalent in theory. However, in practice the latter is far slower. In a third experiment we compare the block-wise update of the QR decomposition using Householder reflections with the column-wise update of the QR decomposition using Givens rotations.

Experiment 3. *We measure the cpu time for constructing the QR decomposition of* 100 *random matrices* $\mathbf{M}$ *of size* $\mathsf{s}_1 = \mathsf{s}_2 = w$ *using Householder re-*

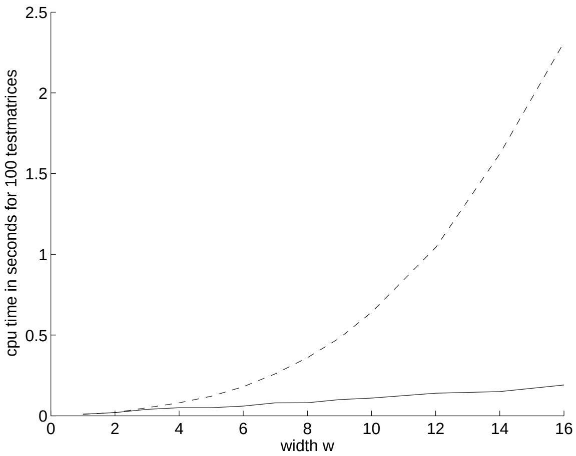


Fig. 3. Experiment 3: Computation time for the QR decomposition of 100 random $2w \times w$ upper trapezoidal matrices $\mathbf{M}$. The solid line represents results gained by using Householder reflections and block-wise updates. The dashed line corresponds to using Givens rotations and column-wise updates.

flections with simultaneous update of the remaining columns as in (37) on the one hand, and using Givens rotations with column-wise update (as described above) on the other hand. See Fig. 3.

VII. Conclusions and generalizations

The symmetric block Lanczos process produces a growing banded matrix $\underline{\mathbf{T}}_n$, which consists of a symmetric block tridiagonal matrix $\mathbf{T}_n$, extended by some additional rows, and a permutation matrix $\mathbf{P}_n$. In a forthcoming paper we will discuss the need for those permutations and deflations in the symmetric block Lanczos process and give details and results for the block MinRes and block SymmLQ algorithms, which require to compute the full QR decomposition $\underline{\mathbf{T}}_n \mathbf{P}_n = \mathbf{Q}_n \underline{\mathbf{R}}_n$. The standard approach for this computation has been an update algorithm based on Givens rotations, a generalization of the well known update algorithm for tridiagonal matrices. It was recommended to compute both $\underline{\mathbf{T}}_n$ and $\mathbf{R}_n$ column by column.

We promote instead a block-wise construction of $\underline{\mathbf{T}}_n$ and a block-wise update algorithm based on Householder reflections for the QR decomposition. It turns out that our QR decomposition is equally accurate as the one based on Givens rotations and that even on a serial computer it is much faster than column-wise updates with Givens rotations.

At least formally our approach can be generalized quickly from the symmetric block Lanczos to the unsymmetric block Lanczos and the block Arnoldi processes, and from the QR decomposition of banded symmetric block tridiagonal matrices to the one of banded unsymmetric block tridiagonal matrices or block Hessenberg matrices as they come up in block QMR and block GMRes, respectively.

REFERENCES

[ALI 00] Aliaga J. I., Boley D. L., Freund R. W., Hernandez V., *A Lanczos-type method for multiple starting vectors*, *Math. Comp.*, vol. 69, p. 1577–1601, 2000.

[BAI 99] Bai Z., Day D., Ye Q., *ABLE: an adaptive block Lanczos method for non-Hermitian eigenvalue problems*, *SIAM J. Matrix Anal. Appl.*, vol. 20, p. 1060–1082 (electronic), 1999, Sparse and structured matrices and their applications (Coeur d'Alene, ID, 1996).

[CUL 74] Cullum J., Donath W. E., A block generalization of the symmetric s-step Lanczos algorithm, Rapport no. 4845, IBM T.J. Watson Research Center, may 1974.

[FRE 97a] Freund R. W., Malhotra M., *A block QMR algorithm for non-Hermitian linear systems with multiple right-hand sides*, *Linear Algebra and Its Applications*, vol. 254, p. 119–157, 1997.

[FRE 97b] Freund R. W., Malhotra M., *A block QMR algorithm for non-Hermitian linear systems with multiple right-hand sides*, *Linear Algebra Appl.*, vol. 254, p. 119–157, 1997.

[FRE 04] Freund R. W., QR Zerlegung im Lanczos Prozess, private note, 2004.

[GOL 77] Golub G. H., Underwood R., The block Lanczos method for computing eigenvalues, *Mathematical Software, III (Proc. Sympos., Math. Res. Center, Univ. Wisconsin, Madison, Wis., 1977)*, p. 361–377. Publ. Math. Res. Center, no. 39, Academic Press, New York, 1977.

[LEH 96] Lehoucq R. B., *The computations of elementary unitary matrices*, *ACM Trans. Math. Software*, vol. 22, p. 393-400, 1996.

[LEW 77] Lewis J., Algorithms for Sparse Matrix Computations, PhD thesis, Stanford University, Standford, CA, 1977.

[O'L 87] O'Leary D. P., *Parallel implementation of the block conjugate gradient algorithm*, *Parallel Computing*, vol. 5, p. 127-139, 1987.

[PAI 75] Paige C. C., Saunders M. A., *Solution of Sparse Indefinite Systems of Linear Equations*, *SIAM J. Numer. Anal.*, vol. 12, p. 617–629, 1975.

[PAR 71] Parlett B. N., *Analysis of algorithms for reflectors in bisectors*, *SIAM Review*, vol. 13, p. 197–208, 1971.

[RUH 79] Ruhe A., *Implementation aspects of band Lanczos algorithms for computation of eigenvalues of large sparse symmetric matrices*, *Math. Comp.*, vol. 33, p. 680–687, 1979.

[UND 75] Underwood R., An iterative block Lanczos method for the solution of large sparse symmetric eigenproblems, PhD thesis, Stanford University, Stanford, CA, 1975.

[WIL 65] Wilkinson J. H., *The Algebraic Eigenvalue Problem*, Oxford University Press, 1965.