Electronic Transactions on Numerical Analysis.
Volume 20, pp. 50-63, 2005.

ISSN 1068-9613.

ETNA
Kent State University
etna@mcs.kent.edu

ON THE USE OF LARGER BULGES IN THE QR ALGORITHM *

DANIEL KRESSNER†

Abstract. The role of larger bulges in the QR algorithm is controversial. Large bulges are infamous for having a strong, negative influence on the convergence of the implicitly shifted QR algorithm. This paper provides a new explanation of this shift blurring effect, by connecting the computation of the first column of the shift polynomial to the notoriously ill-conditioned pole placement problem. To avoid shift blurring, modern variants of the QR algorithm employ chains of tightly coupled tiny bulges instead of one large bulge. It turns out that larger bulges still play a positive role in these variants; a slight increase of the bulge sizes often results in considerable performance improvements.

Key words. QR algorithm, bulges, shift blurring, pole placement

AMS subject classifications. 65F15, 15A18

1. Introduction.

This paper is concerned with the QR algorithm, the most widely used method for computing eigenvalues and invariant subspaces of a real or complex, dense matrix. Having been introduced by Francis [10] and Kublanovskaya [17], in 1961–1962, the QR algorithm contains three ingredients: preliminary reduction to Hessenberg form, QR iterations, and deflations. In this paper, we will mainly focus on the second ingredient and assume that our $n \times n$ matrix A, of interest, has already been reduced to Hessenberg form. An efficient algorithm for achieving this reduction is described in [7], and implemented in LAPACK [1]. We will also assume that A is a real matrix; all obtained results can be extended to the complex case without any difficulties.

An implicitly shifted QR iteration relies on a fortunate choice of m *shifts* $\sigma_1, \dots, \sigma_m \in \mathbb{C}$, yielding the associated *shift polynomial*

$$p(A) := (A - \sigma_1 I_n) \cdot (A - \sigma_2 I_n) \cdots (A - \sigma_m I_n), \tag{1.1}$$

where I_n denotes the $n \times n$ identity matrix. The set of shifts is assumed to be closed under complex conjugation, implying that $p(A)$ is real. If x denotes the first column of $p(A)$, then the QR iteration proceeds with choosing an orthogonal matrix Q_0, e.g., a Householder matrix [11], such that $Q_0^T x$ is a scalar multiple of the first unit vector e_1. Next, A is updated by the orthogonal similarity transformation $A \leftarrow Q_0^T A Q_0$. This destroys the Hessenberg form of A and creates a bulge of size $(m+1) \times (m+1)$, below the subdiagonal in the upper left corner. The QR iteration is completed by reducing A back to Hessenberg form via a sequence of orthogonal similarity transformations. This reduction can be seen as chasing the bulge from the upper left corner down to the bottom right corner along the first subdiagonal of A, a point of view that has been emphasized and extended to other QR-like algorithms by Watkins and Elsner [32].

Early attempts to improve the performance of the QR algorithm focused on using shift polynomials of high degree [3], say $m \geq 64$. Large values for m result in large bulges, which in turn admit the efficient use of medium-order Householder matrices and WY representations [11]. This approach, however, has proved disappointing due to the fact that, the convergence of such a large-bulge multishift QR algorithm is severely affected by roundoff errors [8, 30, 31]. To explain this phenomenon, Watkins [30, 31], investigated the transmission of shifts during QR iterations and established a simple relationship between the shifts and

*Received August 12, 2004. Accepted for publication October 22, 2004. Recommended by G. Ammar. This work is supported by the DFG Research Center "Mathematics for key technologies" (FZT 86) in Berlin.

†kressner@math.tu-berlin.de, Institut für Mathematik MA 4-5, TU Berlin, Str. des 17. Juni 136, D-10623 Berlin, FRG.

ETNA
Kent State University
etna@mcs.kent.edu

the bulges mentioned above, see also Section 2. Numerical experiments give evidence that this relationship is extremely sensitive to perturbations for larger m, in which case the information encoded in the shifts is likely to become completely contaminated by roundoff errors during the QR iteration. As proper shifts are essential for convergence, it can be concluded that this so called *shift blurring* effect, is responsible for the poor behavior of the large-bulge QR algorithm. In Section 3, we will explain the occurrence of these extreme sensitivities via a connection to the notoriously ill-conditioned pole placement problem.

The trouble with shift blurring has led researchers to develop variants of the QR algorithm that still rely on a large number of simultaneous shifts, but chase several tiny bulges instead of one large bulge. This idea has been proposed many times, see [4, 8, 13, 15, 18, 19, 29]. Particularly successful in improving the performance of the QR algorithm on serial machines is a recent variant developed independently by Braman, Byers, and Mathias [4] as well as by Lang [18]. This variant is based on chains of tightly coupled 3×3 bulges, each of which contains two shifts. It achieves high performance by employing matrix-matrix multiplications during the bulge chasing process. In Section 4, we show that the performance of this method can be considerably increased by using slightly larger, say 5×5 or 7×7, bulges. Numerical experiments confirm that this assertion still holds if the recently developed and highly successful early aggressive deflation strategy [5] is used. Hence, if utilized with care, larger bulges still play a role in modern variants of the QR algorithm.

2. Bulges, Bulge Pencils and Shift Blurring.

For the rest of this paper, we assume that the matrix $A \in \mathbb{R}^{n\times n}$ is in upper Hessenberg form, i.e., all entries below the subdiagonal of A are zero. Moreover, A is assumed to be unreduced[1], i.e., all its subdiagonal entries are different from zero. The latter property can always be guaranteed by applying deflations in the course of the QR algorithm.

2.1. Implicitly shifted QR iterations.

In the following, we briefly describe the conventional implicitly shifted QR iteration based on Householder matrices. For a given vector $y \in \mathbb{R}^n$ and an integer $j \leq n$, we will use $H_j(y)$ to denote a Householder matrix, which maps the trailing $n-j$ elements of y to zero, without affecting its leading $j-1$ elements, see [11]. After having chosen m shifts, where typically $m \ll n$, the next step of the QR iteration consists of the update,

$$A \leftarrow H_1(x) \cdot A \cdot H_1(x), \tag{2.1}$$

where x denotes the first column of the shift polynomial defined in (1.1). The following Wilkinson diagram illustrates the shape of A, after this update for $n = 6$ and $m = 2$:

$$\begin{bmatrix} a & a & a & a & a & a \\ b & b & b & a & a & a \\ b & b & b & a & a & a \\ b & b & b & a & a & a \\ 0 & 0 & 0 & a & a & a \\ 0 & 0 & 0 & 0 & a & a \end{bmatrix}. \tag{2.2}$$

The 3×3 submatrix $A(2:4, 1:3)$, whose elements are labeled by b, is called the *bulge*. Here, the *colon notation* $A(i_1 : i_2, j_1 : j_2)$ is used to designate the submatrix of A defined by rows i_1 through i_2 and columns j_1 through j_2. For general $m < n - 1$, the bulge is $(m+1) \times (m+1)$ and resides in the submatrix $A(2 : m+2, 1 : m+1)$. The QR iteration is completed by reducing A back to Hessenberg form, see Algorithm 1. Let us see what happens

[1] Some authors favor the term *proper* instead of unreduced.

ETNA
Kent State University
etna@mcs.kent.edu

Algorithm 1 Reduction to Hessenberg form

Input: A matrix $A \in \mathbb{R}^{n\times n}$.
Output: An orthogonal matrix $Q \in \mathbb{R}^{n\times n}$ such that $A \leftarrow Q^T AQ$ is in upper Hessenberg form.

$Q \leftarrow I_n$
for $j \leftarrow 1,\dots,n-2$ **do**
  $Q \leftarrow Q \cdot H_{j+1}(Ae_j)$
  $A \leftarrow H_{j+1}(Ae_j) \cdot A \cdot H_{j+1}(Ae_j)$
end for

if the first two loops of this algorithm are applied to the matrix A in (2.2):

$$\rightsquigarrow \begin{bmatrix} a & a & a & a & a & a \\ a & a & a & a & a & a \\ 0 & b & b & b & a & a \\ 0 & b & b & b & a & a \\ 0 & b & b & b & a & a \\ 0 & 0 & 0 & 0 & a & a \end{bmatrix} \rightsquigarrow \begin{bmatrix} a & a & a & a & a & a \\ a & a & a & a & a & a \\ 0 & a & a & a & a & a \\ 0 & 0 & b & b & b & a \\ 0 & 0 & b & b & b & a \\ 0 & 0 & b & b & b & a \end{bmatrix}. \tag{2.3}$$

These diagrams illustrate the fact that the reduction of A to Hessenberg form can be seen as, chasing the bulge from the upper left corner down to the bottom right corner, along the first subdiagonal. The rest of Algorithm 1, consists of chasing the bulge off the bottom right corner:

$$\rightsquigarrow \begin{bmatrix} a & a & a & a & a & a \\ a & a & a & a & a & a \\ 0 & a & a & a & a & a \\ 0 & 0 & a & a & a & a \\ 0 & 0 & 0 & b & b & b \\ 0 & 0 & 0 & b & b & b \end{bmatrix} \rightsquigarrow \begin{bmatrix} a & a & a & a & a & a \\ a & a & a & a & a & a \\ 0 & a & a & a & a & a \\ 0 & 0 & a & a & a & a \\ 0 & 0 & 0 & a & a & a \\ 0 & 0 & 0 & 0 & a & a \end{bmatrix}. \tag{2.4}$$

Properly chosen shifts imply that, repeatedly applied QR iterations will let A converge to block triangular form:

$$A \longrightarrow \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}, \quad A_{11} \in \mathbb{R}^{k\times k}, \quad A_{22} \in \mathbb{R}^{(n-k)\times(n-k)}. \tag{2.5}$$

Typically, the shifts are taken to be the eigenvalues of the trailing $m \times m$ principal submatrix of A before each QR iteration. This choice is called *Francis shifts*[2] and ensures local quadratic convergence to a block triangular form (2.5), with $n - k \approx m$ [33]. Another suitable choice is, to use the same set of m shifts throughout all iterations. In this case, the convergence becomes linear and often $k \approx m$ [33]. As soon as A is sufficiently close to (2.5), the (nearly zero) subdiagonal block of A is set to zero and the QR algorithm is continued by applying QR iterations to the diagonal blocks A_{22} and A_{11} separately. This process is called *deflation*; the most common deflation criterion is to set a subdiagonal entry $a_{k+1,k}$ of the Hessenberg matrix A, to zero if it satisfies,

$$|a_{k+1,k}| \leq \mathbf{u} \cdot (|a_{k,k}| + |a_{k+1,k+1}|),$$

where $\mathbf{u}$ denotes the *unit roundoff*. More advanced deflation criteria can be found in [5], see also Section 4.2. Deflations and QR iterations are recursively applied until A has converged

[2]Other terms in use are *Wilkinson shifts* or *generalized Rayleigh quotient shifts*.

ETNA
Kent State University
etna@mcs.kent.edu

to a Wintner-Murnaghan form, also known as real Schur form [11]. For later reference, we note that the unreduced diagonal block which is currently being processed by QR iterations is called the *active submatrix*.

2.2. Bulge pencils. To explain the notion of bulge pencils, assume that the implicitly shifted QR iteration, with $m < n$ shifts, is applied to an unreduced $n \times n$ Hessenberg matrix A and let x denote the first column of the shift polynomial. The *initial bulge pencil* is the matrix pencil $B_0 - \lambda N$, where N is the $(m+1) \times (m+1)$ Jordan block belonging to the eigenvalue zero and

$$B_0 = [x(1:m+1), A(1:m+1:1:m)] = \begin{bmatrix} x_1 & a_{11} & \cdots & a_{1m} \\ x_2 & a_{21} & \ddots & \vdots \\ \vdots & & \ddots & a_{mm} \\ x_{m+1} & 0 & & a_{m+1,m} \end{bmatrix}.$$

There is a surprisingly simple relationship between the shifts and the eigenvalues of this matrix pencil.

THEOREM 2.1 (Watkins [31]). *The shifts $\sigma_1, \ldots, \sigma_m$ are the finite eigenvalues of the initial bulge pencil $B_0 - \lambda N$.*

In Section 2.1, we have seen that during the course of a QR iteration, a bulge is created at the top left corner of A and chased down to the bottom right corner. Let $A^{(j)}$ denote the updated matrix A after the bulge has been chased $j-1$ steps, i.e., $j-1$ loops of Algorithm 1 have been applied to A after the update (2.1). The bulge resides in the submatrix

$$B_j = A^{(j)}(j+1 : j+m+1, j : j+m), \tag{2.6}$$

which is exactly the submatrix designated by the entries b in (2.3).

THEOREM 2.2 (Watkins [31]). *The shifts $\sigma_1, \ldots, \sigma_m$ are the finite eigenvalues of the* jth bulge pencil $B_j - \lambda N$.

Note that the definition of the bulge B_j is only possible for $j \leq n - m - 1$, since otherwise (2.6) refers to entries outside of $A^{(j)}$. Such a situation is displayed in (2.4). This technical issue can be resolved; by adding virtual rows and columns to the matrix $A^{(j)}$, see [31], Theorem 2.2 can be extended to the case $j > n - m - 1$.

2.3. Convergence for large m in finite-precision arithmetic. Theorem 2.2, shows how the shifts are transmitted during QR iterations. In order to achieve quadratic convergence with Francis shifts in finite-precision arithmetic, it is essential that the information contained in these shifts is properly transmitted to the bottom right corner. However, several numerical experiments conducted in [31], show that the finite eigenvalues of the bulge pencils $B_j - \lambda N$ become, as m increases, extremely sensitive to perturbations. Already for $m = 24$, they are often completely swamped with roundoff errors and have no significant digit in common with the intended shifts.

Although no exact relation between the quality of shifts in the bulge pencils and the convergence of the QR algorithm in finite-precision arithmetic, was proven in [31], it is intuitively clear that the described sensitivity may affect the performance severely. Note that, this does not necessarily imply that the QR algorithm does not converge for large m but the convergence is much slower and must be attributed to linear convergence often taking place at the top left corner of A. Figure 2.1 illustrates this phenomenon for matrices generated with the MATLAB [21] command `triu(rand(300),-1)`. As m increases, deflations taking place at the top left corner (signaling linear convergence) start dominating deflations at the bottom

ETNA
Kent State University
etna@mcs.kent.edu

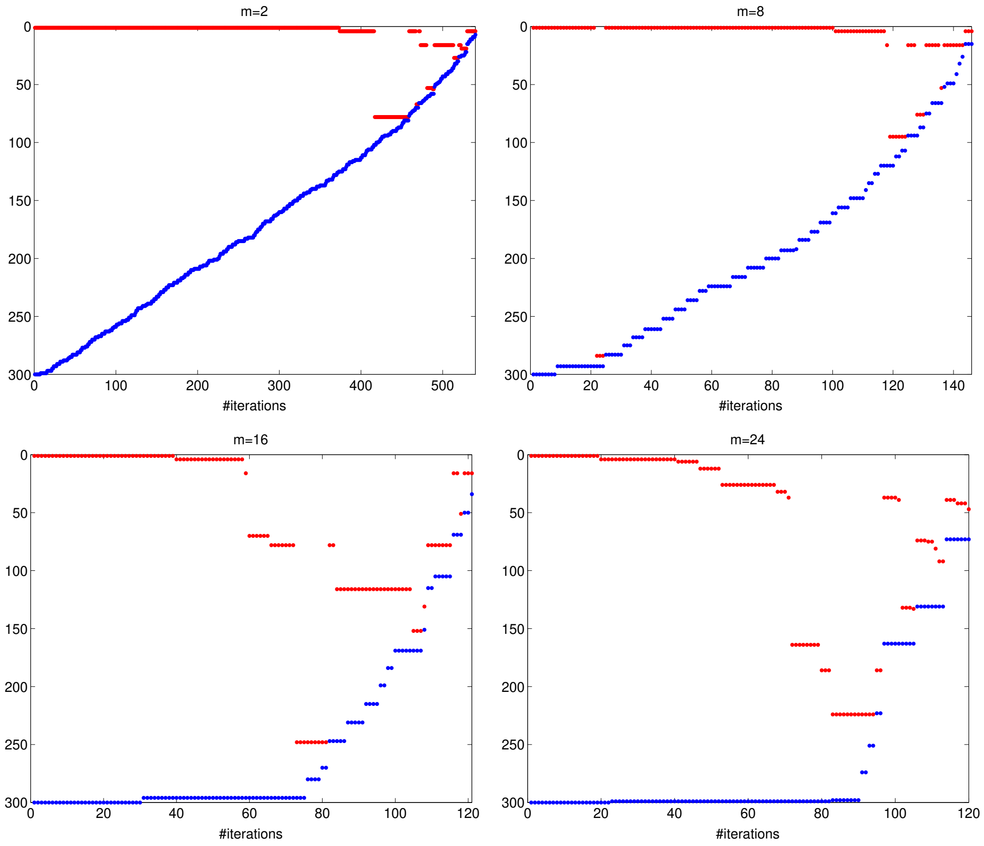

FIG. 2.1. *Lower (red dots) and higher (blue dots) indices of active submatrices that have order larger than m during the multishift QR algorithm with m Francis shifts.*

right corner (signaling quadratic convergence). The QR algorithm requires about $7.4 \cdot 10^8$ flops (floating point operations) for $m = 24$, while it requires only 2.9×10^8 flops for $m = 2$.

Note that the matrix under consideration is known to have extremely ill-conditioned eigenvalues, see [28]. For matrices with fairly well-conditioned eigenvalues – such as those generated by the MATLAB command `hess(rand(300))` – the convergence of the QR algorithm similarly slows down in finite-precision arithmetic as m increases [18, 31]. However, the effect that linear convergence starts dominating quadratic convergence, is not as clearly visible as in Figure 2.1.

3. Connection to Pole Placement. Although convincing numerical experiments for the described shift blurring effects are provided in [31], there has been no explanation why bulge pencils are getting so sensitive as m increases. In this section, we connect the computation of x, the first column of the shift polynomial, to the pole placement problem in systems and control theory, see [25, 26] for a numerical linear algebra oriented introduction.

First, we note that the unreducedness of A implies

$$x_{m+1} = \prod_{i=1}^{m+1} a_{i+1,i} \neq 0.$$

Furthermore, it can be easily shown that neither the implicitly shifted QR iteration nor the statement of Theorem 2.1 is affected if we replace x by a nonzero scalar multiple thereof. Hence, we may assume without loss of generality that x is normalized such that $x_{m+1} = 1$. By applying a simple equivalence transformations to the initial bulge pencil $B_0 - \lambda N$, Theorem 2.1 shows that the shifts $\sigma_1, \dots, \sigma_m$ are the eigenvalues of the matrix

$$\begin{aligned} C &= A(1:m,1:m) - a_{m+1,m}x(1:m)e_m^T \\ &= \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1,m-1} & a_{1m} - a_{m+1,m}x_1 \\ a_{21} & a_{22} & \dots & a_{2,m-1} & a_{2m} - a_{m+1,m}x_2 \\ 0 & a_{32} & \dots & a_{3,m-1} & a_{3m} - a_{m+1,m}x_3 \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & a_{m,m-1} & a_{mm} - a_{m+1,m}x_m \end{bmatrix}. \end{aligned} \tag{3.1}$$

Next, consider the single-input control system

$$\dot{z}(t) = A(1:m,1:m)^T z(t) - (a_{m+1,m}e_m)u(t) \tag{3.2}$$

with state vector $z(\cdot)$ and input $u(\cdot)$. The linear state feedback $u(t) = x(1:m)^T z(t)$ yields the closed-loop matrix C^T, where C is defined as in (3.1). Hence, the feedback vector $x(1:m)$ places the poles of the open loop system (3.2) to $\sigma_1, \dots, \sigma_m$. Since $x(1:m)$ is uniquely defined by this property, we obtain the following connection:

Any pole placement algorithm for single-input systems is a suitable method for computing a multiple of the first column of the shift polynomial; and vice versa.

To some extent, this connection has already been used by A.Varga for designing a multi-shift pole placement algorithm [27]. A not-so-serious application is the expression of the QL iteration [9], a permuted version of the QR iteration, in three lines of MATLAB code, using functions of the MATLAB Control Toolbox [20]:

```
s = eig(A(1:m,1:m));
x = acker(A(n-m+1:n,n-m+1:n)', A(n-m,n-m+1)*eye(m,1), s);
A = ctrbf(A, [zeros(1,n-m-1) 1 x]', []);
```

An exceedingly more serious consequence is caused by observing that placing a large number of poles in a single-input problem is often very ill-conditioned [12, 14], in the sense that,the poles of the closed loop system are very sensitive to perturbations in the input data. The nature of this ill-conditioning was analyzed in detail by Mehrmann and Xu [22, 23]. Assume that the input data of the pole placement problem – $B = A(1:m,1:m)^T, b = a_{m+1,m}e_m$, and $\sigma_1, \dots, \sigma_m$ – are perturbed by sufficiently small perturbations $\triangle B$, $\triangle b$ and $\triangle\sigma_1, \dots, \triangle\sigma_m$. Set

$$\varepsilon = \max\{\|[\triangle B, \triangle b]\|, |\triangle\sigma_1|, \dots, |\triangle\sigma_m|\},$$

and let $\hat{f}$ be the feedback vector defined by the perturbed problem. Then, it was shown in [23, Thm. 1.1] that the eigenvalues $\hat{\sigma}_1, \dots, \hat{\sigma}_m$ of $B - b\hat{f}^T$ satisfy,

$$|\hat{\sigma}_i - \sigma_i| \leq \left(1 + \|G\|_2\|G^{-1}\|_2\sqrt{1 + \|\hat{f}\|_2}\right)\varepsilon + \mathcal{O}(\varepsilon^2), \tag{3.3}$$

where G is the eigenvector matrix of the closed loop matrix $C^T = A + bf^T$, normalized such that all columns of G have unit norm. Although (3.3) is only an upper bound, numerical

ETNA
Kent State University
etna@mcs.kent.edu

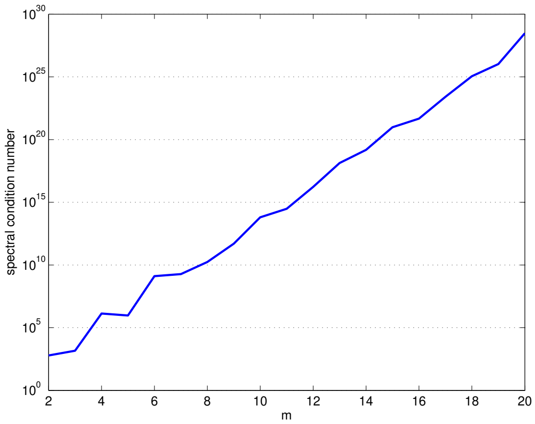

FIG. 3.1. *Condition number of the eigenvector matrix G of the closed loop matrix C^T.*

experiments in [22, 23] suggest that (3.3) catches the qualitative behavior of the maximal eigenvalue error rather well.

The presence of the condition number $\|G\|_2 \|G^{-1}\|_2$ in (3.3), is particularly worrying. Often, this term grows rapidly with m, even exponential growth can be proven for some cases [22, Ex. 1]. Indeed, such an effect can be observed in the QR algorithm. For this purpose, we constructed the closed loop matrix C^T, as defined in (3.1), for a matrix A generated by the MATLAB command `hess(rand(250))`. Figure 3.1 displays the condition number of the corresponding eigenvector matrix G for $m = 2, \ldots, 15$ Francis shifts, clearly exhibiting exponential growth.

The described connection to the notoriously ill-conditioned pole placement problem yields an explanation for the sensitivity of the initial bulge pencil. However, it does not explain the sensitivity of the bulge pencils $B_1 - \lambda N$, $B_2 - \lambda N$, ... On the other hand, there is little hope that the shifts, once destroyed because of the sensitivity of $B_0 - \lambda N$, recover during the bulge chasing process although this event sometimes occurs in practice [31].

4. Tightly Coupled Tiny Bulges. Chasing chains of tightly coupled tiny bulges instead of one large bulge,is an approach which avoids shift blurring but is still capable to benefit from a large number of Francis shifts. To describe this approach, let m, denote the number of simultaneous shifts to be used in each QR iteration, and let n_s denote the number of shifts contained in each bulge. It is assumed that m is an integer multiple of n_s. To avoid shift blurring effects we use tiny values for n_s, say $n_s \in [2, 6]$.

Our algorithm performs an implicitly shifted QR iteration with m Francis shifts on an unreduced Hessenberg matrix A and consists of three stages, which are described in more detail below. First, a chain of m/n_s tightly coupled bulges is bulge-by-bulge introduced in the top left corner A. Second, the whole chain at once is chased down along the subdiagonal until the bottom bulge reaches the bottom right corner of A. Finally, all bulges are bulge-by-bulge chased off this corner.

Note that, the only aspect in which our algorithm extends the algorithms described in [4, 18] is that the latter are restricted to 3×3 bulges, i.e., $n_s = 2$.

Introducing a chain of bulges. Given a set of m Francis shifts, we partition this set into subsets $\Sigma_1, \Sigma_2, \ldots, \Sigma_{m/n_s}$. Each Σ_j contains n_s shifts and is closed under complex conjugation. We apply the implicitly shifted QR iteration with the shifts contained in Σ_1 and interrupt the bulge chasing process as soon as the bottom right corner of the bulge touches

ETNA
Kent State University
etna@mcs.kent.edu

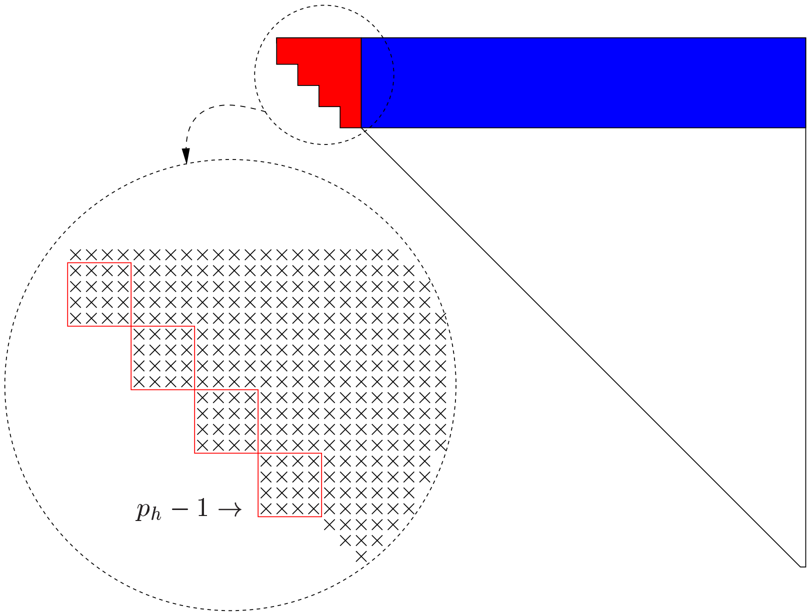

FIG. 4.1. *Introducing a chain of $m/n_s = 4$ tightly coupled bulges, each of which contains $n_s = 3$ shifts.*

the $(p_h - 1, p_h)$ subdiagonal entry of A, where $p_h = (m/n_s)(n_s + 1) + 1$. Next, the bulge belonging to Σ_2 is introduced and chased so that,its bottom right corner is at the $(p_h - n_s - 2, p_h - n_s - 1)$ subdiagonal entry. This process is continued until all m/n_s bulges are stringed like pearls on the subdiagonal of the submatrix $A(1 : p_h, 1 : p_h)$, see Figure 4.1. Note that only this submatrix (painted red in Figure 4.1) must be updated during the bulge chasing process. To update the remaining part of A (painted blue), all employed orthogonal transformations are accumulated into a $p_h \times p_h$ matrix U. This enables us to use matrix-matrix multiplications:

$$A(1 : p_h, (p_h + 1) : n) \leftarrow U^T \cdot A(1 : p_h, (p_h + 1) : n).$$

Chasing a chain of bulges. Suppose that a chain of bulges resides on the subdiagonal of the submatrix $A(p_l : p_h, p_l : p_h)$, where $p_h = p_l + (n_s + 1)m/n_s$. In the beginning, we have $p_l = 1$ and $p_h = (m/n_s)(n_s + 1) + 1$ but we will now subsequently increase these values by chasing the complete chain. To move the chain to the submatrix $A(p_l + k : p_h + k, p_l + k : p_h + k)$, each individual bulge is chased k steps, as depicted in Figure 4.2. This is done in bottom-to-top order,so that,no bulges have to cross each other.

Again only a submatrix, namely $A(p_l : p_h + k, p_l : p_h + k)$, must be updated during the bulge chasing process. To update the rest of the matrix, we accumulate all transformations in an orthogonal matrix U of order $((n_s+1)m/n_s+k+1)$ and use matrix-matrix multiplications:

$$\begin{aligned} A(p_l : p_h + k, (p_h + 1) : n) &\leftarrow U^T \cdot A(p_l : p_h + k, (p_h + 1) : n), \\ A(1 : p_l - 1, p_l : p_h + k) &\leftarrow A(1 : p_l - 1, p_l : p_h + k) \cdot U. \end{aligned}$$

Note that U has a particular block structure that can be exploited to increase the efficiency of

ETNA
Kent State University
etna@mcs.kent.edu

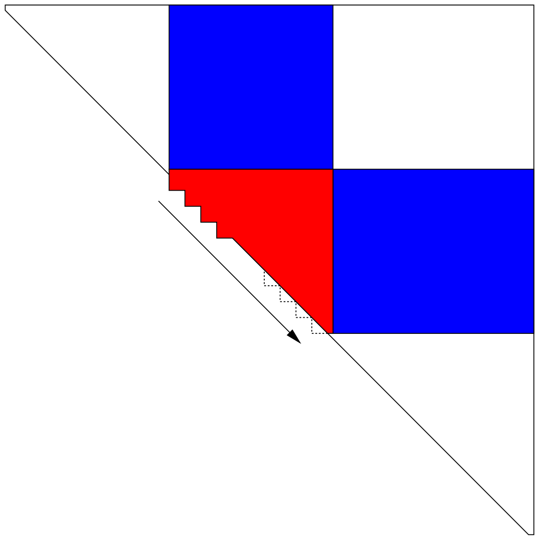

FIG. 4.2. *Chasing a chain of $m/n_s = 4$ tightly coupled bulges.*

these multiplications:

$$U = \begin{bmatrix} 1 & 0 & 0 \\ 0 & U_{11} & U_{12} \\ 0 & U_{21} & U_{22} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \square & \triangleright \\ 0 & \triangleleft & \square \end{bmatrix}, \tag{4.1}$$

i.e., the matrices $U_{12} \in \mathbb{R}^{l_1 \times l_1}$ and $U_{21} \in \mathbb{R}^{l_2 \times l_2}$, where $l_1 = (m/n_s)(n_s + 1) - n_s$ and $l_2 = k + n_s$, are lower and upper triangular, respectively. There is even more structure present, as illustrated in Figure 4.3. It is, however, difficult to take advantage of this extra banded structure using level 3 BLAS [6].

The rare event of a zero subdiagonal entry between two consecutive bulges, during the bulge chasing process is called a "vigilant" deflation [29]. Such a deflation causes a severe loss of information,if bulges are chased from above through this zero subdiagonal entry. This can be avoided by reintroducing the bulges in the row,in which the zero appears using essentially the same method that has been used for introducing bulges [4]. Note that it is not necessary to take care of vigilant deflations caused by small non-zero subdiagonal entries [30].

Getting rid of a chain of bulges. Once the bottom bulge of the chain has reached the bottom right corner of A, the whole chain is bulge-by-bulge chased off this corner, similarly to the introduction of bulges at the top left corner of A.

4.1. Numerical experiments. The described QR algorithm based on chains of tightly coupled tiny bulges has been implemented in a Fortran 77 routine called `MTTQR`. Although this routine generally requires more flops than standard implementations of the QR algorithm [1], it can be expected that this extra cost is more than compensated by the fact that `MTTQR` facilitates level 3 BLAS for a large part of the computation, see also [4].

Note that,the packing density of the bulges is getting higher as n_s, the number of shifts per bulge, increases. For example, a 16×16 principal submatrix of A may either contain 10 shifts distributed over five 3×3 bulges or it may contain 12 shifts distributed over three 5×5

ETNA
Kent State University
etna@mcs.kent.edu

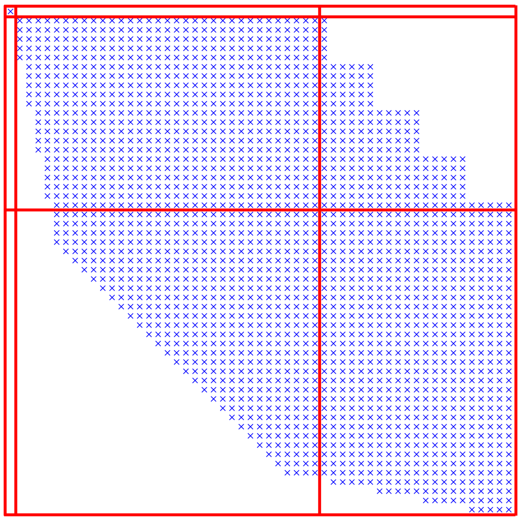

FIG. 4.3. *Structure of the transformation matrix* U *for chasing a chain of* $m/n_s = 5$ *bulges, each of which contains* $n_s = 4$ *shifts,* $k = 30$ *steps.*

Matrix name	n	Description
OLM1000	1000	Olmstead model
TUB1000	1000	tubular reactor model
TOLS1090	1090	Tolosa matrix
TOLSBAL	1090	balanced Tolosa matrix
RDB1250	1250	reaction-diffusion Brusselator model, $L = 0.5$
RDB1250L	1250	reaction-diffusion Brusselator model, $L = 1$
BWM2000	2000	Brusselator wave model in chemical reaction
OLM2000	2000	Olmstead model
DW2048	2048	square dielectric waveguide
RDB2048	2048	reaction-diffusion Brusselator model, $L = 0.5$
RDB2048L	2048	reaction-diffusion Brusselator model, $L = 1$
PDE2961	2961	partial differential equation

TABLE 4.1
Subset of matrices from the test matrix collection [2].

bulges. In either case, essentially the same amount of operations is necessary to chase the chain of bulges from top to bottom. Hence, if shift blurring does not cause problems, using larger values for n_s can improve the efficiency of `MTTQR`.

To verify these statements, we applied `MTTQR` to a subset of real $n \times n$ matrices from the test matrix collection [2], see also Table 4.1. Note that TOLSBAL is the matrix obtained after balancing [24] has been applied to the highly unbalanced matrix TOLS1090. For the parameters m (number of shifts in each iteration) and k (number of steps a chain of bulges is chased before off-diagonal parts are updated), we followed the recommendations given in [4]:

$$m = \begin{cases} 60, & \text{if } 1000 \le n < 2000, \\ 120, & \text{if } 2000 \le n < 2500, \\ 156, & \text{if } 2500 \le n < 3000, \end{cases}$$

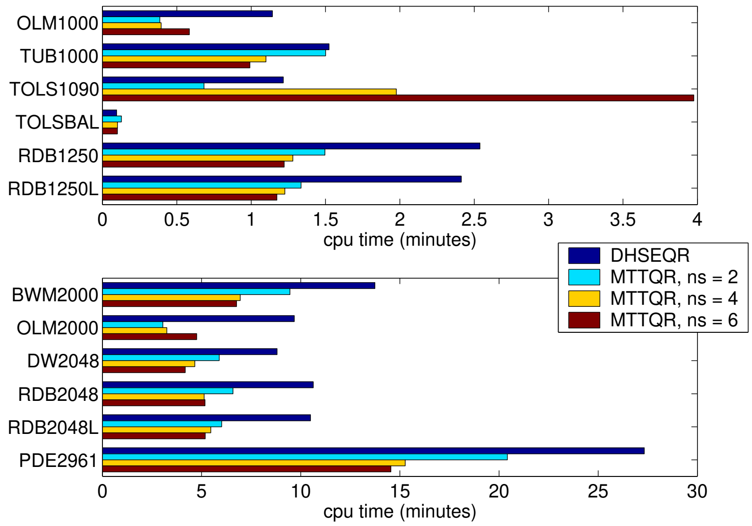

FIG. 4.4. *Execution times for* DHSEQR *and* MTTQR*, the tiny-bulge multishift QR algorithm with* $n_s \in \{2, 4, 6\}$ *shifts per bulge, applied to matrices from the test matrix collection [2].*

and $k = 3/2 \cdot m - 2$.

Our numerical experiments were performed on an IBM Power3 based SMP system with four 375 Mhz Power3 Processors and 4 gigabytes of memory. The Fortran 77 routines are based on the BLAS kernels provided by IBM's machine-specific optimized Fortran library ESSL. We used the XL Fortran compiler with optimization level 3. Matrices,were always stored in an array,with leading dimension slightly larger than the number of rows to avoid unnecessary cache conflicts.

Figure 4.4 compares the cpu times required by MTTQR with those required by DHSEQR, LAPACK's implementation of the QR algorithm. ¿From these times, we may conclude that MTTQR with $n_s = 2$ shifts per bulge requires considerably less time than DHSEQR for all considered matrices except TOLSBAL and TUB1000. For TOLSBAL, MTTQR consumes 34% more time, which seems to be due to the fact that the QR algorithm converges so quickly that the overhead in MTTQR dominates any performance improvements gained by using matrix-matrix multiplications . For TUB1000, we obtain a performance improvement of only 1.5%, which is much less than for the other matrices, where this figure ranges from 25% up to 69%. Increasing n_s from 2 to 4 often leads to even further speedups. A notable exception is TOLS1090, where MTTQR requires 190% more time if $n_s = 4$ instead of $n_s = 2$ is used. We believe that,this behavior can be attributed to the poor balancing of this matrix, which seems to amplify shift blurring effects. The highest improvements can be obtained for TUB1000 and BWM2000, where MTTQR requires 27% less time if $n_s = 4$ instead of $n_s = 2$ is used. Increasing n_s further, from 4 to 6, can lead to additional (albeit often insignificant) speedups; but it also raises the risk of shift blurring.

4.2. Influence of aggressive early deflation. We repeated the numerical experiments from the previous section,by combining aggressive early deflation with the QR algorithm based on chains of tightly coupled bulges in a Fortran 77 routine called ATTQR. Aggressive early deflation is a highly successful deflation strategy that takes advantage of matrix perturbations outside of the commonly considered subdiagonal entries, see [5] for more details.

ETNA
Kent State University
etna@mcs.kent.edu

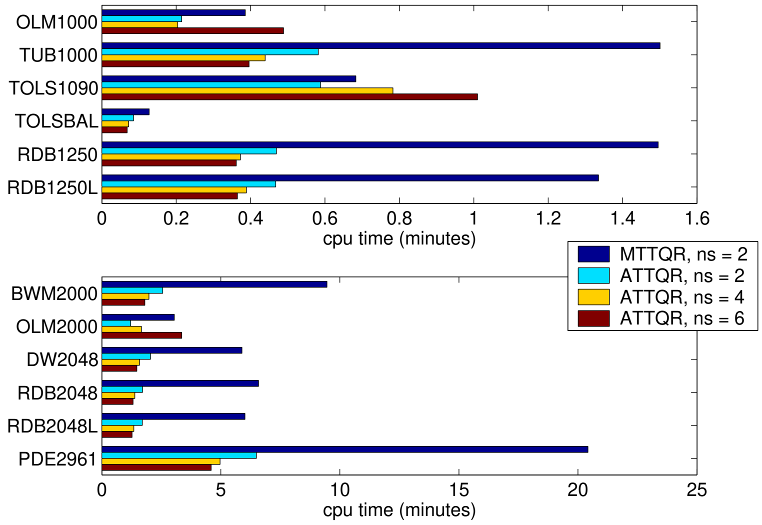

FIG. 4.5. *Execution times for* ATTQR*, the tiny-bulge multishift QR algorithm with aggressive early deflation and* $n_s \in \{2, 4, 6\}$ *shifts per bulge, applied to matrices from the test matrix collection [2].*

Our choice of parameters for ATTQR is based on the recommendations given in [5]:

$$m = \begin{cases} 96, & \text{if } 1000 \leq n < 2000, \\ 120, & \text{if } 2000 \leq n < 2500, \\ 180, & \text{if } 2500 \leq n < 3000, \end{cases}$$

$k = 3/2 \cdot m - 2$, and $w = 3/2 \cdot m$ (size of deflation window).

The cpu times displayed in Figure 4.5 show that the use of aggressive early deflation results in substantial improvements. The gained cpu time savings (for the case that $n_s = 2$ shifts per bulge are used) range from 14% for the TOLS1090 matrix up to 74% for the RDB2048 matrix. Again, increasing n_s from 2 to 4 leads to even further speedups, except for TOLS1090 and OLM2000. For all other matrices, the gained savings range from 5% up to 24%.

5. Conclusion. The conclusion that can be drawn from this paper is two-edged. On the one hand, the explained connection to the pole placement problem confirms the well-known wisdom that large bulges severely deteriorate the convergence of the QR algorithm. On the other hand, we have shown that the use of slightly larger bulges can still have a positive effect on the performance of modern variants of the QR algorithm. A good compromise seems to be made by 5×5 bulges.

6. Final Remarks and Acknowledgments. The work presented in this article is based on preliminary results derived in [16]. The Fortran 77 routines MTTQR and ATTQR, used in the numerical experiments, are available on request from the author.

The author is greatly indebted to Volker Mehrmann for illuminating discussions and to Ralph Byers for providing Fortran implementations of the algorithms described in [4, 5]; these implementations laid the basis for the Fortran routines used in this paper. The numerical experiments in Section 4 were performed using facilities of the High Performance Computing Center North (HPC2N) in Umeå, Sweden.

REFERENCES

[1] E. ANDERSON, Z. BAI, C. BISCHOF, S. BLACKFORD, J. W. DEMMEL, J. J. DONGARRA, J. DU CROZ, A. GREENBAUM, S. HAMMARLING, A. MCKENNEY, AND D. C. SORENSEN, *LAPACK Users' Guide*, SIAM, Philadelphia, PA, third edition, 1999.

[2] Z. BAI, D. DAY, J. W. DEMMEL, AND J. J. DONGARRA, *A test matrix collection for non-Hermitian eigenvalue problems (release 1.0)*, Technical Report CS-97-355, Department of Computer Science, University of Tennessee, Knoxville, TN, USA, March 1997, also available online from http://math.nist.gov/MatrixMarket.

[3] Z. BAI AND J. W. DEMMEL, *On a block implementation of the Hessenberg multishift QR iterations,* Internat. J. High Speed Comput., 1 (1989), pp. 97–112.

[4] K. BRAMAN, R. BYERS, AND R. MATHIAS, *The multishift QR algorithm. I. Maintaining well-focused shifts and level 3 performance*, SIAM J. Matrix Anal. Appl., 23 (2002), No. 4, pp. 929–947.

[5] K. BRAMAN, R. BYERS, AND R. MATHIAS, *The multishift QR algorithm, II, Aggressive early deflation*, SIAM J. Matrix Anal. Appl., 23 (2002), No. 4, pp. 948–973.

[6] J. J. DONGARRA, J. DU CROZ, I. S. DUFF, AND S. HAMMARLING, *A set of level 3 basic linear algebra subprograms,* ACM Trans. Math. Software, 16 (1990), pp. 1–17.

[7] J. J. DONGARRA, D. C. SORENSEN, AND S. J. HAMMARLING, *Block reduction of matrices to condensed forms for eigenvalue computations,* J. Comput. Appl. Math., 27 (1989), No. 1-2, pp. 215–227, reprinted in *Parallel algorithms for numerical linear algebra*, pp. 215–227, North-Holland, Amsterdam, 1990.

[8] A. A. DUBRULLE, *The multishift QR algorithm–is it worth the trouble?* TR 6320-3558, IBM Scientific Center, Palo Alto, CA, 1991.

[9] A. A. DUBRULLE, R. S. MARTIN, AND J. H. WILKINSON, *The implicit QL algorithm,* Numerische Mathematik, 12 (1968), pp. 377–383, also in [34, pp.241–248].

[10] J. G. F. FRANCIS, *The QR transformation, parts I and II,* Computer Journal, 4 (1961), pp. 265–271, 4 (1962), pp. 332–345.

[11] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, Johns Hopkins University Press, Baltimore, MD, third edition, 1996.

[12] C. HE, A. J. LAUB, AND V. MEHRMANN, *Placing plenty of poles is pretty preposterous,* Preprint SPC 95-17, Forschergruppe 'Scientific Parallel Computing', Fakultät für Mathematik, TU Chemnitz-Zwickau, 1995.

[13] G. HENRY, D. S. WATKINS, AND J. J. DONGARRA, *A parallel implementation of the nonsymmetric QR algorithm for distributed memory architectures,* SIAM J. Sci. Comput., 24 (2002), No. 1, pp. 284–311.

[14] N. J. HIGHAM, M. KONSTANTINOV, V. MEHRMANN, AND P. PETKOV, *Sensitivity of computational control problems,* Numerical Analysis Report No. 424, Manchester Centre for Computational Mathematics, Manchester, England, February 2003, to appear in IEEE Control Systems Magazine,

[15] L. KAUFMAN, *A parallel QR algorithm for the symmetric tridiagonal eigenvalue problem,* Journal of Parallel and Distributed Computing, 3 (1994), pp. 429–434.

[16] D. KRESSNER, *Numerical Methods and Software for General and Structured Eigenvalue Problems*, PhD thesis, TU Berlin, Institut für Mathematik, Berlin, Germany, 2004.

[17] V. N. KUBLANOVSKAYA, *On some algorithms for the solution of the complete eigenvalue problem,* Zhurnal Vychislitelnoi Matematiki i Matematicheskoi Fiziki, 1 (1961), pp. 555–570.

[18] B. LANG, *Effiziente Orthogonaltransformationen bei der Eigen- und Singulärwertzerlegung*, Habilitationsschrift, 1997.

[19] B. LANG, *Using level* 3 *BLAS in rotation-based algorithms,* SIAM J. Sci. Comput., 19 (1998), No. 2, pp. 626–634.

[20] *The* MATLAB *Control Toolbox, Version 5*, The MathWorks, Inc., Cochituate Place, 24 Prime Park Way, Natick, MA, 2000.

[21] MATLAB *Version 6.5*, The MathWorks, Inc., Cochituate Place, 24 Prime Park Way, Natick, MA, 2002.

[22] V. MEHRMANN AND H. XU, *An analysis of the pole placement problem, I, The single-input case,* Electron. Trans. Numer. Anal., 4 (1996), pp. 89–105, http://etna.mcs.kent.edu/vol.4.1996/pp89-105.dir/pp89-105.pdf

[23] V. MEHRMANN AND H. XU, *Choosing poles so that the single-input pole placement problem is well conditioned,* SIAM J. Matrix Anal. Appl., 19 (1998), No. 3, pp. 664–681.

[24] B. N. PARLETT AND C. REINSCH, *Balancing a matrix for calculation of eigenvalues and eigenvectors,* Numerische Mathematik, 13 (1969), pp. 293–304, also in [34, pp.315–326].

[25] P. H. PETKOV, N. D. CHRISTOV, AND M. M. KONSTANTINOV, *Computational Methods for Linear Control Systems*, Prentice-Hall, Hertfordshire, UK, 1991.

[26] P. VAN DOOREN, *Numerical Linear Algebra for Signal, Systems and Control*, Draft notes prepared for the Graduate School in Systems and Control, 2003.

[27] A. VARGA, *A multishift Hessenberg method for pole assignment of single-input systems,* IEEE Trans. Automat. Control, 41 (1996), No. 12, pp. 1795–1799.

[28] D. VISWANATH AND L. N. TREFETHEN, *Condition numbers of random triangular matrices,* SIAM J. Matrix Anal. Appl., 19 (1998), No. 2, pp. 564–581.

[29] D. S. WATKINS, *Shifting strategies for the parallel QR algorithm,* SIAM J. Sci. Comput., 15 (1994), No. 4,

pp. 953–958.

[30] D. S. WATKINS, *Forward stability and transmission of shifts in the QR algorithm,* SIAM J. Matrix Anal. Appl., 16 (1995), No. 2, pp. 469–487.

[31] D. S. WATKINS, *The transmission of shifts and shift blurring in the QR algorithm,* Linear Algebra Appl., 241/243 (1996), pp. 877–896.

[32] D. S. WATKINS AND L. ELSNER, *Chasing algorithms for the eigenvalue problem,* SIAM J. Matrix Anal. Appl., 12 (1991), No. 2, pp. 374–384.

[33] D. S. WATKINS AND L. ELSNER, *Convergence of algorithms of decomposition type for the eigenvalue problem,* Linear Algebra Appl., 143 (1991), pp. 19–47.

[34] J. H. WILKINSON AND C. REINSCH, *Handbook for Automatic Computation, Vol. II Linear Algebra*, Springer-Verlag, New York, 1971.