# An Implementation of a Divide and Conquer Algorithm for the Unitary Eigenproblem

*G. S. Ammar*
*L. Reichel*
*Dan Sorensen*

**CRPC-TR90057**
**June, 1990**

Center for Research on Parallel Computation
Rice University
P.O. Box 1892
Houston, TX 77251-1892

# An implementation of a divide and conquer algorithm for the unitary eigenproblem *

G.S. Ammar
Northern Illinois University

L. Reichel
University of Kentucky

D.C. Sorensen
Rice University

**Abstract.** We present a FORTRAN implementation of a divide-and-conquer method for computing the spectral resolution of a unitary upper Hessenberg matrix $H$. Any such matrix $H$ of order n, normalized so that its subdiagonal elements are nonnegative, can be written as a a product of $n-1$ Givens reflectors and a diagonal matrix. This representation, which we refer to as the *Schur parametric form* of $H$, arises naturally in applications such as in signal processing and in the computation of Gauss-Szegő quadrature rules. Our programs utilize the Schur parameterization to compute the spectral decomposition of $H$ without explicitly forming the elements of $H$. If only the eigenvalues and first components of the eigenvectors are desired, as in the applications mentioned above, the algorithm requires only $O(n^2)$ arithmetic operations. Experimental results presented indicate that the algorithm is reliable and competitive with the general QR algorithm applied to this problem. Moreover, the algorithm can be easily adapted for parallel implementation.

Categories and Subject Descriptors: G.1.3 [Numerical Analysis]: Numerical Linear Algebra; G.4 [Mathematics of Computing]: Mathematical Software.

General Terms: Algorithms.

Additional Key Words and Phrases: Unitary eigenvalue problem, fast algorithm.

Authors' Addresses: G.S. Ammar, Northern Illinois University, Department of Mathematical Sciences, DeKalb, IL 60115; L. Reichel, University of Kentucky, Department of Mathematics, Lexington, KY 40506; D.C. Sorensen, Rice University, Department of Mathematical Sciences, Houston, TX 77251.

## 1 Introduction

There is considerable interest in recently developed divide-and-conquer algorithms for the symmetric eigenproblem because they are competitive on sequential computers and well suited to parallel implementation; see [3], [4], [9], [11]. Divide-and-conquer techniques can also be applied to the solution of eigenvalue problems for unitary matrices [2], [6], [7]. The present paper describes an implementation of the algorithm introduced in [6], [7].

Any unitary upper Hessenberg matrix $H$ of order $n$ with nonnegative subdiagonal elements can be written

*Research supported by NSF grant DMS-9002884, NSF cooperative agreement CCR-8809615 and IBM Bergen Scientific Centre.

in a unique way as the product

$$H = H(\gamma_1, \gamma_2, \ldots, \gamma_n) = G_1 G_2 \ldots G_{n-1} \tilde{G}_n, \tag{1.1}$$

where the $G_k \in \mathbf{C}^{n \times n}$, $1 \leq k < n$, are Givens reflectors,

$$G_k = \begin{bmatrix} I_{k-1} & & & \\ & -\gamma_k & \sigma_k & \\ & \sigma_k & \bar{\gamma}_k & \\ & & & I_{n-k-1} \end{bmatrix}, \quad \gamma_k \in \mathbf{C},\ \sigma_k \in \mathbf{R},\ \sigma_k \geq 0,\ |\gamma_k|^2 + \sigma_k^2 = 1, \tag{1.2a}$$

and $\tilde{G}_n$ is the diagonal matrix

$$\tilde{G}_n := \begin{bmatrix} I_{n-1} & \\ & -\gamma_n \end{bmatrix}, \quad \gamma_n \in \mathbf{C},\ |\gamma_n| = 1. \tag{1.2b}$$

Here $I_j$ denotes the identity matrix of order $j$, and $\bar{\gamma}_j$ denotes the complex conjugate of $\gamma_j$. We refer to the $\gamma_j$, $1 \leq j \leq n$, as the *Schur parameters* of $H$, the $\sigma_j$, $1 \leq j < n$, as the *complementary parameters* of $H$, and the representation (1.1) as the *Schur parametric form* of $H$. Note that the complementary parameters are the subdiagonal elements of $H$. If a unitary upper Hessenberg matrix $H$ is given componentwise, then it is straightforward to determine its Schur parametric form (see, e.g., [7]).

Observe that, although the complementary parameters are mathematically determined by the Schur parameters, $\sigma_j$ cannot be accurately recovered from $\gamma_j$ numerically when $|\gamma_j|$ is close to one. In fact, if $|\gamma_j| = 1$, $\sigma_j$ could be any nonnegative machine number less than the square root of machine precision. We therefore retain the Schur parameters and the complementary parameters during computations to avoid numerical instability.

The divide-and-conquer algorithm we implement obtains the spectral resolution of a unitary upper Hessenberg matrix $H$ with nonnegative subdiagonal elements by manipulating Schur parameters and complementary parameters instead of matrix elements. This makes a low operation count possible, and also preserves structure in the problem. For example, our implementation numerically preserves the property that the eigenvalues of $H$ must lie on the unit circle.

We remark that in applications of the unitary eigenproblem in signal processing (described in [1], [13]), as well as in certain applications of computing Gauss-Szegő quadrature rules (discussed in [7]), a unitary upper Hessenberg matrix $H$ is *given* in Schur parametric form. In the applications to signal processing discussed in [1], [13], the Schur parametric form of a unitary upper Hessenberg matrix $H$ is determined from an autocorrelation matrix for the signal, so that the eigenvalues of $H$ are the estimated frequencies of the dominant harmonics of the signal up to a scaling factor that depends on the sampling frequency. The square of the first components of the normalized eigenvectors of $H$ give the amplitudes of the estimated dominant harmonics. The applications

to signal processing are closely related to the computation of Gauss-Szegő quadrature rules. A discussion of the mathematical properties of Gauss-Szegő quadrature rules can be found in [5], [8, Chapter 4], [10].

We outline how the spectral resolution of $H$ is obtained from the spectral resolutions of two subproblems. Details are described in [7]. The algorithm is based on the fact that a complex Givens reflector $G_s$ is diagonally unitarily equivalent with a real Givens reflector, and that the latter can be written as a Householder transformation. Introduce

$$\gamma_s' := \begin{cases} \gamma_s/|\gamma_s|, & \gamma_s \neq 0, \\ 1, & \gamma_s = 0. \end{cases} \tag{1.3}$$

Then $|\gamma_s'| = 1$ and

$$\begin{bmatrix} I_{s-1} & & \\ & \bar{\gamma}_s' & \\ & & I_{n-s} \end{bmatrix} G_s \begin{bmatrix} I_s & & \\ & \gamma_s' & \\ & & I_{n-s-1} \end{bmatrix} = \begin{bmatrix} I_{s-1} & & & \\ & -|\gamma_s| & \sigma_s & \\ & \sigma_s & |\gamma_s| & \\ & & & I_{n-s-1} \end{bmatrix}. \tag{1.4}$$

The real Givens reflector on the right of (1.4) can be written as a Householder transformation $I - 2ww^*$, where

$$w := \omega_s e_s + \omega_{s+1} e_{s+1} \in \mathbf{R}^n, \tag{1.5a}$$

$$\omega_s := 2^{-1/2}(1 + |\gamma_s|)^{1/2}, \tag{1.5b}$$

$$\omega_{s+1} := -2^{-1/2}\sigma_s(1 + |\gamma_s|)^{-1/2}. \tag{1.5c}$$

Throughout this paper $*$ denotes transposition and complex conjugation, and $e_j$ denotes the $j^{\text{th}}$ axis vector of appropriate length. We now can write (1.1) as

$$H = \begin{bmatrix} H_1 & \\ & I_{n-s} \end{bmatrix} (I - 2ww^*) \begin{bmatrix} I_s & \\ & H_2 \end{bmatrix}, \tag{1.6}$$

where

$$H_1 : := H(\gamma_1, \gamma_2, \ldots, \gamma_{s-1}, -\gamma_s') \in \mathbf{C}^{s \times s}, \tag{1.7a}$$

$$H_2 : := H(\bar{\gamma}_s'\gamma_{s+1}, \bar{\gamma}_s'\gamma_{s+2}, \ldots, \bar{\gamma}_s'\gamma_n) \in \mathbf{C}^{(n-s)\times(n-s)}. \tag{1.7b}$$

Assume that the unitary matrices $W_k$ and diagonal matrices $\Lambda_k$ in the spectral resolutions of the submatrices $H_k$ are explicitly known; i.e., assume that we know the factorizations

$$H_k = W_k \Lambda_k W_k^*, \qquad k = 1, 2. \tag{1.8}$$

Define the block-diagonal matrices

$$\tilde{H} := \begin{bmatrix} H_1 & \\ & H_2 \end{bmatrix}, \quad \tilde{W} := \begin{bmatrix} W_1 & \\ & W_2 \end{bmatrix}, \tag{1.9a}$$

$$\tilde{\Lambda} = \text{diag}[\lambda_1, \lambda_2, \ldots, \lambda_n] := \begin{bmatrix} \Lambda_1 & \\ & \Lambda_2 \end{bmatrix}, \tag{1.9b}$$

and the vector

$$z = [\zeta_j]_{j=1}^n := \begin{bmatrix} W_1^* e_s \omega_s \\ \bar{\Lambda}_2 W_2^* e_1 \omega_{s+1} \end{bmatrix}. \tag{1.9c}$$

Substitution of (1.9) into (1.6) yields

$$H = \tilde{W}\tilde{\Lambda}(I - 2zz^*)\tilde{W}^* = \tilde{H} - 2\tilde{W}\tilde{\Lambda}zz^*\tilde{W}^* ; \tag{1.10}$$

i.e., $H$ can be written as a rank-one modification of the unitary matrix $\tilde{H}$. We can now describe how the eigenvalues of $H$ can be obtained from the spectral resolution of $\tilde{H}$. In view of (1.9)–(1.10) the characteristic polynomial $\chi(\lambda) := \det(\lambda I - H)$ can be written as

$$\chi(\lambda) = \det(\lambda I - \tilde{H} + 2\tilde{W}\tilde{\Lambda}zz^*\tilde{W}^*) = \det(\lambda I - \tilde{H})\left(1 + 2z^*(\lambda I - \tilde{\Lambda})^{-1}\tilde{\Lambda}z\right) .$$

Thus, the eigenvalues of $H$ include the zeros of the spectral function

$$\phi(\lambda) := 1 + 2z^*(\lambda I - \tilde{\Lambda})^{-1}\tilde{\Lambda}z ,$$

which we can write as

$$\phi(\lambda) = \sum_{j=1}^{n} |\zeta_j|^2 \frac{\lambda + \lambda_j}{\lambda - \lambda_j} , \tag{1.11}$$

where we have used the fact that $z^*z = 1$. Substitution of

$$\lambda = \exp(i\theta), \quad \lambda_j = \exp(i\theta_j), \quad -\pi < \theta, \theta_j \le \pi, \quad 1 \le j \le n,$$

into (1.11) yields

$$\Phi(\theta) := \sum_{j=1}^{n} |\zeta_j|^2 \cot\left(\frac{\theta - \theta_j}{2}\right) = -i\phi(\lambda), \tag{1.12}$$

where $i := \sqrt{-1}$.

If the $\theta_j$ are pairwise distinct and all the $\zeta_j$ are nonvanishing, then $\Phi(\theta)$ has $n$ distinct zeros $\theta_j'$, and the eigenvalues of $H$, which are given by $\lambda_j' = \exp(i\theta_j')$ $(1 \le j \le n)$, strictly interlace the eigenvalues of $\tilde{H}$ on the unit circle. A normalized eigenvector of $H$ associated with $\lambda_j'$ can then be computed as

$$v_j := v(\lambda_j')/\|v(\lambda_j')\|_2, \quad \lambda_j' = \exp(i\theta_j'), \tag{1.13}$$

where

$$v(\lambda) := \tilde{W}(I - \tilde{\Lambda}^*\lambda)^{-1}z ; \tag{1.14}$$

and $\| \ \|_2$ denotes the Euclidean length of a vector. See [7] for details. The expression (1.9c) is slightly modified from the definition of the vector $z$ in [7] so that the eigenvector formula (1.14) is simplified.

If some $\zeta_j$ vanishes or if the $\theta_j$ are not pairwise distinct, then $\Phi(\theta)$ will have fewer than $n$ zeros (the number of terms in (1.11) and (1.12) will be effectively less than $n$). In such cases, the eigenvalues of $H$ that cannot be determined from zeros of $\Phi$ are also eigenvalues of $\tilde{H}$, and the corresponding eigenvectors of $H$ are easily determined. Moreover, if some $\zeta_j$ is tiny or if two singular points $\theta_j$ and $\theta_k$ of $\Phi$ are very close, then a zero of $\Phi$ may be very close to a singular point, and the corresponding eigenvector computed by (1.13)–(1.14) might not be accurate due to cancellation of significant digits. These difficulties can be circumvented by deflation, which is discussed in Section 2. Deflation also reduces the computational work required to solve an eigenproblem because there are fewer roots of $\Phi$ to find. In Section 5 we give examples of matrices that lead to massive deflation and substantial reductions in computational work. We remark that also in the divide-and-conquer method for the *symmetric* eigenproblem certain matrices give rise to massive deflation and a corresponding reduction in computational work; see [3] and [4] for a discussion.

Formula (1.14) requires that the differences $\theta'_j - \theta_k$ be known with high relative accuracy, otherwise the eigenvectors determined by (1.13)–(1.14) might not be orthogonal. In Section 3 we outline our implementation of a rootfinder for (1.12) that uses differences $\theta - \theta_j$ as variables. The computed differences corresponding to a zero of (1.12) are then substituted into (1.14). This avoids the loss of significant digits that can take place if we first compute a root $\theta'_j$ of (1.12) and then form the differences $\theta'_j - \theta_k$.

In our implementation of the divide-and-conquer algorithm, we subdivide the original eigenproblem until only eigenproblems of orders 2 and 1 remain. These eigenproblems are solved analytically. The known spectral resolutions of smaller eigenproblems are then used to determine spectral resolutions of larger ones, as described above, in a recursive manner until the original eigenproblem has been solved. Implementation details on the subdivision strategy are considered in Section 4. Section 5 contains computed examples.

Note that the eigenvalues of $H$ can be determined from those of $\tilde{H}$ and the vector $z$ which is determined by the bottom row of $W_1$ and the top row of $W_2$. The divide-and-conquer algorithm can therefore be restricted if the full spectral resolution of $H$ is not desired. We refer to the ***partial spectral resolution*** of $H$ as the eigenvalues and the first and last components of the normalized eigenvectors of $H$. Note that the partial spectral resolutions of $H_1$ and $H_2$ determine that of $H$. The first and last components of $v_j$ in (1.13) can be obtained without explicitly forming $v(\lambda'_j)$ by using the fact that $\|v(\lambda'_j)\|_2 = (\frac{1}{2}\Phi'(\theta'_j))^{1/2}$; see [7, (2.19)].

This paper describes two FORTRAN subroutines UDC and UDCP. The routine UDC computes the spectral resolution of $H$, while UDCP determines the partial spectral resolution of $H$, where $H$ is given in terms of its Schur parameters and complementary parameters. The latter subroutine requires only $O(n^2)$ arithmetic operations and $O(n)$ storage locations, and finds application in certain signal processing algorithms ([1], [13]), and in the construction of Gauss-Szegő quadrature rules ([7]). The FORTRAN subroutines of this paper are the first carefully designed codes that use the structure of unitary upper Hessenberg matrices. The subroutines use level one BLAS described in [12].

## 2 Deflation

As described in the introduction, the formulas (1.12)–(1.14) can yield poor accuracy in finite precision arithmetic if either of the quantities

$$\epsilon' := \min_{\substack{1\le j,k\le n\\ j\ne k}} |\theta_j - \theta_k| \tag{2.1}$$

$$\epsilon'' := \min_{1\le j\le n} |\zeta_j| \tag{2.2}$$

is "tiny". On the other hand, if $\epsilon'$ or $\epsilon''$ is of very small magnitude, then eigenpairs of $H$ can be determined without using formulas (1.12)–(1.14), and the size of the problem (i.e., the number of terms in (1.12)) can be reduced. This is called deflation.

Assume for the moment that a component $\zeta_j$ of $z$ vanishes. Then (1.10) yields

$$H\tilde{W}e_j = \tilde{W}\tilde{\Lambda}(I - 2zz^*)e_j = \tilde{W}e_j\lambda_j\,; \tag{2.3}$$

i.e., $\{\lambda_j, \tilde{W}e_j\}$ is an eigenpair of $H$. In the computer program, we accept $\{\lambda_j, \tilde{W}e_j\}$ as an eigenpair of $H$ when $|\zeta_j|$ is sufficiently small. The term in (1.11) and (1.12) corresponding to $\lambda_j$ is then omitted. We refer to this procedure as *deflation of type 1.*

A formula closely related to (2.3) can be used to determine an eigenpair if two diagonal entries of $\tilde{\Lambda} = \mathrm{diag}[\lambda_1, \lambda_2, \ldots, \lambda_n]$ are close. Assume that $\lambda_1 \approx \lambda_2$, and introduce the Givens reflector

$$G = \begin{bmatrix} -\gamma & \sigma & \\ \sigma & \bar{\gamma} & \\ & & I_{n-2} \end{bmatrix}, \quad \gamma := -\frac{\bar{\zeta}_1\zeta_2}{\rho|\zeta_2|}, \quad \sigma := \frac{|\zeta_2|}{\rho}, \quad \rho := \left(|\zeta_1|^2 + |\zeta_2|^2\right)^{1/2}. \tag{2.4}$$

Then $(Gz)^*e_2 = 0$, and from (1.10) we obtain

$$\begin{aligned} H\tilde{W}G^*e_2 &= \tilde{W}G^*(G\tilde{\Lambda}G^*)\,(I - Gz(Gz)^*)\,e_2 \\ &= WG^*(G\tilde{\Lambda}G^*)e_2 \approx WG^*\hat{\Lambda}e_2 \\ &= WG^*e_2\tfrac{\lambda_1\sigma^2+\lambda_2|\gamma|^2}{|\lambda_1\sigma^2+\lambda_2|\gamma|^2|}, \end{aligned} \tag{2.5}$$

where

$$\hat{\Lambda} := \mathrm{diag}\left(\frac{\lambda_1|\gamma|^2 + \lambda_2\sigma^2}{|\lambda_1|\gamma|^2 + \lambda_2\sigma^2|}, \frac{\lambda_1\sigma^2 + \lambda_2|\gamma|^2}{|\lambda_1\sigma^2 + \lambda_2|\gamma|^2|}, \lambda_3, \lambda_4, \ldots, \lambda_n\right).$$

Observe that the diagonal entries of $\hat{\Lambda}$ are on the unit circle as they should be, and that the error introduced through replacing $\tilde{\Lambda}$ with $\hat{\Lambda}$ is easily bounded. Using $\lambda_j = \exp(i\theta_j)$ yields the inequalities for $j = 1, 2$,

$$\left|e_j^*(G\tilde{\Lambda}G^* - \hat{\Lambda})e_{3-j}\right| = |\gamma\sigma(\lambda_1 - \lambda_2)| = 2\sigma|\gamma|\left|\sin\left(\frac{\theta_1 - \theta_2}{2}\right)\right|, \tag{2.6}$$

$$\left|e_j^*(G\tilde{\Lambda}G^* - \hat{\Lambda})e_j\right| \le 4\sigma^2|\gamma|^2\sin^2\left(\frac{\theta_1 - \theta_2}{2}\right). \tag{2.7}$$

Formulas (2.5)–(2.7) suggest that we may replace $G\tilde{\Lambda}G^*$ by $\hat{\Lambda}$ and accept $\left\{\frac{\lambda_1\sigma^2+\lambda_2|\gamma|^2}{|\lambda_1\sigma^2+\lambda_2|\gamma|^2|}, WG^*e_2\right\}$ as an eigenpair if

$$\epsilon_{12} := 2\sigma|\gamma| \left|\sin\left(\frac{\theta_1-\theta_2}{2}\right)\right| \tag{2.8}$$

is sufficiently small. In such a case we replace $\tilde{\Lambda}$ by $\hat{\Lambda}$, $\tilde{W}$ by $\tilde{W}G^*$, and $z$ by $Gz$ in the computer program, and the term corresponding to the modified $\lambda_2$ is omitted from (1.12). We refer to this procedure as *deflation of type 2*. Analogously with the definition of $\epsilon_{12}$ in (2.8), we can define quantities $\epsilon_{jk}$ for all $j \neq k$.

In our implementation, we choose a deflation tolerance $\epsilon$ to be a modest multiple of machine precision. Deflation of type 1 is then performed for each $j$ such that $|\zeta_j| \leq \epsilon$. This guarantees that $\epsilon'' > \epsilon$. Let $m$ denote the number of terms in the modified $\Phi$ after type 1 deflation is performed. The $m$ singular points of $\Phi$ are then ordered so that

$$-\pi < \theta_1 \leq \theta_2 \leq \ldots \leq \theta_m \leq \pi\,,$$

and the quantities $\epsilon_{j,j+1}$, $1 \leq j < m$, and $\epsilon_{m1}$ are calculated. If one of these quantities is less than or equal to $\epsilon$, then deflation of type two is performed, $m$ is reduced by one, and we calculate a new set of quantities $\epsilon_{jk}$. This process is repeated until type 2 deflation does not occur. In this way we can assure that two singular points of $\Phi$ are not too close.

The above formulas for deflation are applied before zeros of $\Phi(\theta)$ are computed and eigenvectors are determined by (1.14). Analogous formulas for deflation in the divide-and-conquer method for the symmetric eigenproblem are described in [4].

## 3 Implementation of the rootfinder

Let $\Phi(\theta)$ denote the spectral function obtained after deflation. In particular, all the $\zeta_j$ defining $\Phi$ are nonvanishing, the singular points $\theta_j$ of $\Phi$ are pairwise distinct, and the zeros $\theta'_j$ strictly interlace the $\theta_j$. The eigenpairs of $H$ remaining after deflation are determined by finding the zeros of $\Phi$ and using the formulas (1.13)–(1.14).

Let $\theta_1$ and $\theta_2$ be two distinct adjacent singular points of $\Phi$. In order to determine the unique zero $\theta'_1$ in the interval $(\theta_1, \theta_2)$, we first evaluate $\Phi$ at $\theta_{mid} := \frac{1}{2}(\theta_1 + \theta_2)$. If $\Phi(\theta_{mid}) > 0$, then $\theta'_1$ lies in the interval $(\theta_1, \theta_{mid})$ because $\Phi$ is monotonically increasing between its singular points [7]. We then form the differences $\Delta_{j1} := \theta_j - \theta_1$ for all $j$, and represent $\theta$ by the new variable $\delta := \theta - \theta_1$. We determine $\delta'$ corresponding to the zero $\theta'_1$ of $\Phi$ by the monotonically and quadratically convergent rootfinder described in [6], supplied with appropriate stopping criteria. The eigenvector corresponding to $\lambda'_1 := \exp(i\theta'_1)$ is obtained by substituting $\delta' - \Delta_{j1}$ for $\theta - \theta_j$ in (1.14). The introduction of the variable $\delta$ is important because it allows us to avoid the loss of significant digits that would result in the computation of the eigenvector using the value of $\theta'_1$ when $|\theta'_1 - \theta_1|$ is small. If $\Phi(\theta_{mid}) < 0$, then we form and use the differences $\Delta_{j2} := \theta_j - \theta_2$ for all $j$ and the variable $\delta := \theta - \theta_2$ in a similar fashion. The other zeros of $\Phi$ are computed analogously.

## 4 Generation of subproblems

Given a unitary upper Hessenberg matrix of order $n$ in Schur parametric form

$$H^{(n)} = H(\gamma_1, \gamma_2, \ldots, \gamma_n),$$

we apply formulas (1.3)–(1.7) to obtain subproblems of orders $n-2$ and 2,

$$\begin{aligned} H_1^{(n-2)} &= H(\gamma_1, \gamma_2, \ldots, \gamma_{n-3}, -\gamma'_{n-2}), \\ H_1^{(2)} &= H(\bar{\gamma}'_{n-2}\gamma_{n-1}, \bar{\gamma}'_{n-2}, \gamma'_n). \end{aligned}$$

The spectral resolution of $H^{(n)}$ can be determined from the matrices $H_1^{(n-2)}$ and $H_1^{(2)}$, and the parameters $\{|\gamma_{n-2}|, \sigma_{n-2}\}$. The matrix $H_1^{(n-2)}$ is subdivided further in an analogous manner. When $n$ is even we obtain the unitary matrices $H_1^{(2)}, H_2^{(2)}, \ldots, H_{n/2}^{(2)}$ of order 2, and when $n$ is odd we obtain the unitary matrices $H_1^{(2)}, H_2^{(2)}, \ldots, H_{(n-1)/2}^{(2)}, H_{(n+1)/2}^{(1)}$, all of which are of order 2, except the matrix $H_{(n+1)/2}^{(1)}$ which is of order 1. Thus for $n$ odd we have to determine the eigenvalues and eigenvectors of the matrix

$$\text{block-diagonal}\left[H_{(n+1)/2}^{(1)}, H_{(n-1)/2}^{(2)}, H_{(n-3)/2}^{(2)}, \ldots, H_1^{(2)}\right]. \tag{4.1}$$

The spectral resolution of each block can be determined in closed form; see below. We use spectral resolutions of adjacent $2 \times 2$ or $1 \times 1$ diagonal blocks from top to bottom in order to compute the spectral resolutions of eigenproblems of order 4 or 3. Next we use the spectral resolutions of the diagonal blocks so obtained from bottom up to determine spectral resolutions of matrices of larger order. We proceed in this manner until the spectral resolution of $H^{(n)}$ has been found. The computation of spectral resolutions of subproblems in the order indicated makes all subproblems to be used together have roughly the same size. This maintains load balancing in the parallel setting and also reduces the computational work in the serial case. The computations are organized analogously if only the partial spectral resolution of $H^{(n)}$ is desired.

The computation of the spectral resolution of the matrix (4.1) requires the determination of eigenvalues and eigenvectors of Givens reflectors $G_j$ and of a matrix of the form $G_{n-1}\tilde{G}_n$. The computations have to be organized so as to avoid cancellation of significant digits. For instance, consider the Givens reflector

$$G := \begin{bmatrix} -\gamma & \sigma \\ \sigma & \bar{\gamma} \end{bmatrix}, \quad \gamma \in \mathbb{C}, \ \sigma \geq 0, \ |\gamma|^2 + \sigma^2 = 1,$$

and assume that $\text{Re}(\gamma)^2 + \sigma^2 > 0$. Then $\lambda_1 = \sqrt{\text{Re}(\gamma)^2 + \sigma^2} - i\text{Im}(\gamma)$ is an eigenvalue of $G$ and an associated unnormalized eigenvector is given by $[\alpha, \beta]^T$, where we in computations use the formulas

$$\begin{aligned} &\beta := 1; \quad \alpha := \sigma / \left(\text{Re}(\gamma) + \sqrt{\text{Re}(\gamma)^2 + \sigma^2}\right), \quad \text{Re}(\gamma) \geq 0, \\ &\alpha := 1; \quad \beta := \sigma / \left(-\text{Re}(\gamma) + \sqrt{\text{Re}(\gamma)^2 + \sigma^2}\right), \quad \text{Re}(\gamma) < 0. \end{aligned} \tag{4.2}$$

The other eigenvalue of $G$ is given by $\lambda_2 = -\sqrt{\text{Re}(\gamma)^2 + \sigma^2} - i\text{Im}(\gamma)$, and $[\beta, -\alpha]^T$ is an associated eigenvector. If $\text{Re}(\gamma) = \sigma^2 = 0$, then $-\gamma$ is a double eigenvalue of $G$, and we choose the axis vectors as eigenvectors. Formulas similar to (4.2) are used for the computation of the eigenvectors of the matrix of the form $G_{n-1}\tilde{G}_n$.

## 5 Computed examples

We now present some experimental results to compare the performance of our FORTRAN subroutines UDC and UDCP with that of the subroutines COMQR2 and COMQR of EISPACK [14]. The subroutine UDC determines the (full) spectral resolution of a unitary upper Hessenberg matrix $H$ in Schur parametric form by the divide-and-conquer method described, and UDCP computes the partial spectral resolution of $H$. Both subroutines UDC and UDCP are supplied in a single precision and a double precision version. The EISPACK subroutine COMQR2 is an implementation of the QR algorithm for computing the spectral resolution of a general upper Hessenberg matrix, and the subroutine COMQR computes only the eigenvalues of a general upper Hessenberg matrix.

We compare the accuracy of single precision versions of UDC and COMQR2 as follows. A set of Schur parameters $\{\gamma_j\}_{j=1}^{n}$ and complementary parameters $\{\sigma_j\}_{j=1}^{n-1}$ that correspond to a unitary upper Hessenberg matrix $H$ are input to the single precision version of UDC, which produces the spectral resolution of $H$. The parameters are then used to explicitly calculate the entries of the matrix $H$, and this matrix is input to the single precision version of COMQR2. The eigenvalues produced by COMQR2 are divided by their moduli to insure that they lie on the unit circle. The input parameters are also used to form the matrix $H$ in double precision arithmetic; this matrix is input to the double precision version of COMQR2, and the resulting eigenvalues are again forced onto the unit circle. The eigenvalue matrix $\Lambda = \text{diag}[\exp(i\theta_j)]_{j=1}^{n}$ and the eigenvector matrix $W$ produced by double precision COMQR2 in this manner are assumed to be exact in order to compare the accuracy of the single precision UDC and COMQR2 subroutines. In the tables, $\hat{\Lambda} = \text{diag}[\exp(i\hat{\theta}_j)]_{j=1}^{n}$ and $\hat{W}$ denote the eigenvalue and eigenvector matrices that are produced by either single precision subroutine. The accuracy of the eigenvalues is measured by $\max_{1\le j\le n}|\theta_j - \hat{\theta}_j|$. The accuracy of the eigenvectors is represented by $\|\,|W| - |\hat{W}|\,\|$, where $|W|$ denotes the matrix whose entries are the moduli of the entries of $W$. (The moduli are compared to account for arbitrary unimodular scalings of the columns of $W$.) The quantity $\|\hat{W}^*\hat{W} - I\|$ measures the orthogonality of the computed eigenvectors. The ***eigenresidual*** $\|H\hat{W} - \hat{W}\hat{\Lambda}\|$ provides another measure of the accuracy in the computed spectral resolution. The matrix norm $\|\ \|$ denotes the Frobenius norm; i.e., for $A = [a_{jk}]_{j,k=1}^{n}$ we have $\|A\| := (\sum_{j,k=1}^{n}|a_{jk}|^2)^{1/2}$. Our accuracy experiments were carried out on a Sequent Symmetry computer, where single precision and double precision arithmetic roughly correspond to 7 and 15 significant decimal digits, respectively. In all of our experiments, the deflation tolerance, described in Section 2, is equal to $10^{-6}$. The accuracy of the partial spectral resolution produced by UDCP was essentially the same as that produced by UDC.

*Example 1:* In our first experiment, we input Schur parameters $\gamma_j = \rho_j \exp(i\alpha_j)$, $1 \le j \le n$, where the arguments $\alpha_j$ and the moduli $\rho_j$ are randomly generated according to uniform distributions in $[0, 2\pi]$ and $[0, 1]$, respectively. The modulus of $\gamma_n$ is then set equal to one, so that the corresponding matrix $H$ is unitary. The single precision subroutines UDC and COMQR2 are used to compute the spectral resolution of $H$, and both sets of computed results are compared with the results of the double precision version of COMQR2. We performed this experiment

|  | $\|\|H\hat{W} - \hat{W}\hat{\Lambda}\|\|$ |  | $\|\|\hat{W}^*\hat{W} - I\|\|$ |  | $\max_{1\leq j\leq n} \|\theta_j - \hat{\theta}_j\|$ |  | $\|\| \|W\| - \|\hat{W}\| \|\|$ |  |
|---|---|---|---|---|---|---|---|---|
|  | udc | comqr2 | udc | comqr2 | udc | comqr2 | udc | comqr2 |
| n = 10 |  |  |  |  |  |  |  |  |
| avg: | .37E-05 | .17E-05 | .28E-05 | .29E-05 | .25E-06 | .51E-06 | .14E-05 | .19E-05 |
| max: | .39E-04 | .27E-05 | .33E-04 | .12E-04 | .14E-05 | .89E-06 | .17E-04 | .11E-04 |
| better: | 42 |  | 71 |  | 98 |  | 81 |  |
| same: | 96 |  | 97 |  | 100 |  | 98 |  |
| n = 15 |  |  |  |  |  |  |  |  |
| avg: | .64E-05 | .26E-05 | .48E-05 | .74E-05 | .35E-06 | .74E-06 | .30E-05 | .45E-05 |
| max: | .34E-04 | .36E-05 | .25E-04 | .61E-04 | .60E-05 | .13E-05 | .24E-04 | .16E-04 |
| better: | 20 |  | 70 |  | 98 |  | 77 |  |
| same: | 97 |  | 100 |  | 99 |  | 100 |  |
| n = 20 |  |  |  |  |  |  |  |  |
| avg: | .79E-05 | .35E-05 | .65E-05 | .14E-04 | .35E-06 | .90E-06 | .60E-05 | .93E-05 |
| max: | .35E-04 | .46E-05 | .44E-04 | .75E-04 | .37E-05 | .17E-05 | .37E-04 | .69E-04 |
| better: | 15 |  | 78 |  | 97 |  | 78 |  |
| same: | 99 |  | 100 |  | 100 |  | 100 |  |
| n = 25 |  |  |  |  |  |  |  |  |
| avg: | .10E-04 | .44E-05 | .97E-05 | .26E-04 | .39E-06 | .11E-05 | .94E-05 | .13E-04 |
| max: | .59E-04 | .55E-05 | .86E-04 | .40E-03 | .24E-05 | .19E-05 | .68E-04 | .14E-03 |
| better: | 4 |  | 78 |  | 96 |  | 70 |  |
| same: | 99 |  | 100 |  | 100 |  | 99 |  |
| n = 30 |  |  |  |  |  |  |  |  |
| avg: | .11E-04 | .54E-05 | .95E-05 | .33E-04 | .45E-06 | .12E-05 | .13E-04 | .30E-04 |
| max: | .46E-04 | .67E-05 | .47E-04 | .60E-03 | .41E-05 | .20E-05 | .19E-03 | .15E-02 |
| better: | 11 |  | 85 |  | 94 |  | 78 |  |
| same: | 100 |  | 100 |  | 100 |  | 99 |  |
| n = 40 |  |  |  |  |  |  |  |  |
| avg: | .22E-04 | .73E-05 | .46E-04 | .12E-03 | .58E-06 | .15E-05 | .42E-03 | .19E-03 |
| max: | .12E-03 | .90E-05 | .17E-02 | .34E-02 | .62E-05 | .27E-05 | .35E-01 | .12E-01 |
| better: | 6 |  | 73 |  | 97 |  | 52 |  |
| same: | 99 |  | 98 |  | 100 |  | 93 |  |
| n = 50 |  |  |  |  |  |  |  |  |
| avg: | .27E-04 | .94E-05 | .28E-03 | .98E-04 | .69E-06 | .18E-05 | .27E-03 | .60E-04 |
| max: | .13E-03 | .11E-04 | .23E-01 | .11E-02 | .16E-04 | .29E-05 | .17E-01 | .56E-03 |
| better: | 5 |  | 78 |  | 96 |  | 46 |  |
| same: | 98 |  | 98 |  | 100 |  | 93 |  |

Table 1: Results of 100 runs

| $n = 10$ | | | | | | |
|---|---|---|---|---|---|---|
| | udc deflations | | $\max_{1 \le j \le n} \lvert\theta_j - \hat{\theta}_j\rvert$ | | $\lVert\, \lvert W\rvert - \lvert\hat{W}\rvert \,\rVert$ | |
| $\rho$ | type 1 | type2 | udc | comqr2 | udc | comqr2 |
| .900000 | 0 | 0 | .24E-06 | .89E-06 | .79E-06 | .97E-06 |
| .990000 | 0 | 0 | .24E-06 | .95E-06 | .86E-06 | .15E-05 |
| .999000 | 2 | 0 | .12E-06 | .54E-06 | .70E-06 | .87E-06 |
| .999900 | 4 | 0 | .24E-06 | .36E-06 | .25E-05 | .25E-06 |
| .999990 | 6 | 0 | .24E-06 | .24E-06 | .15E-05 | .16E-06 |
| 1.00000 | 20 | 0 | .24E-06 | .00E+00 | .00E+00 | .00E+00 |

| $n = 20$ | | | | | | |
|---|---|---|---|---|---|---|
| | udc deflations | | eigenvalue error | | eigenvector error | |
| $\rho$ | type 1 | type2 | udc | comqr2 | udc | comqr2 |
| .900000 | 2 | 2 | .48E-06 | .60E-06 | .90E-04 | .39E-03 |
| .990000 | 8 | 0 | .24E-06 | .89E-06 | .13E-05 | .26E-05 |
| .999000 | 18 | 0 | .24E-06 | .63E-06 | .15E-05 | .16E-04 |
| .999900 | 25 | 1 | .24E-06 | * | .33E-05 | * |
| .999990 | 28 | 2 | .24E-06 | * | .15E-05 | * |
| 1.00000 | 59 | 0 | .24E-06 | .00E+00 | .00E+00 | .00E+00 |

| $n = 30$ | | | | | | |
|---|---|---|---|---|---|---|
| | udc deflations | | eigenvalue error | | eigenvector error | |
| $\rho$ | type 1 | type2 | udc | comqr2 | udc | comqr2 |
| .900000 | 6 | 0 | .36E-06 | .95E-06 | .83E-04 | .99E-04 |
| .990000 | 25 | 2 | .24E-06 | .15E-05 | .47E-05 | .47E-05 |
| .999000 | 39 | 0 | .48E-06 | * | .53E-05 | * |
| .999900 | 50 | 0 | .24E-06 | * | .55E-05 | * |
| .999990 | 54 | 3 | .24E-06 | * | .11E-03 | * |
| 1.00000 | 104 | 0 | .24E-06 | .00E+00 | .00E+00 | .00E+00 |

Table 2: Nearly diagonal unitary matrices

100 times for various values of $n$. Table 1 shows the average and maximum values of the indicated quantities over these 100 runs. Also displayed are the number of times that udc produced smaller values than COMQR2 (labeled *better*) and the number of times the value produced by UDC was within 10 times the value produced by COMQR2 (labeled *same*). Note that the maximum error in the arguments of the computed eigenvalues is consistently smaller for UDC than for COMQR2.

*Example 2:* Table 2 displays the results of some experiments in which the Schur parameters are generated as in Example 1, except that the moduli of the Schur parameters are restricted to the interval $[\rho, 1]$. Observe that as $\rho$ tends to one, the complementary parameters tend to zero, and if $\rho = 1$, then the matrix $H$ is diagonal. The number of deflations for UDC, which is displayed in the table, increases as $\rho$ tends to one. The asterisks in Table 2 indicate problems for which the single precision version of COMQR2 produced an overflow exception. We did not invistage the reason for these failures.

*Example 3:* Our third example is designed so that the Hessenberg matrix $H$ has multiple or nearly multiple eigenvalues by making it a nearly block diagonal matrix with identical blocks. More precisely, let $n = pk$. We generate $p - 1$ Schur parameters $\gamma_j$ as in Example 1, and set $\sigma_p = \epsilon$ and $\gamma_p = (1 - \epsilon^2)^{1/2}$. The remaining

2 blocks of order 10; $n = 20$

| | udc deflations | | $\|H\hat{W} - \hat{W}\hat{\Lambda}\|$ | | $\|\hat{W}^*\hat{W} - I\|$ | | $\max_{1\leq j\leq n} \|\theta_j - \hat{\theta}_j\|$ | | $\|\,\|W\| - \|\hat{W}\|\,\|$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| $\epsilon$ | Type 1 | Type 2 | udc | comqr2 | udc | comqr2 | udc | comqr2 | udc | comqr2 |
| 1E-1 | 0 | 0 | .54E-05 | .40E-05 | .21E-04 | .15E-02 | .24E-06 | .48E-06 | .66E-04 | .22E-03 |
| 5E-2 | 0 | 0 | .14E-04 | .32E-05 | .54E-04 | .79E-03 | .24E-06 | .57E-06 | .22E-03 | .62E-03 |
| 1E-2 | 0 | 0 | .63E-05 | .30E-05 | .22E-03 | .10E-01 | .24E-06 | .48E-06 | .15E-02 | .21E-02 |
| 5E-3 | 0 | 0 | .72E-05 | .37E-05 | .92E-03 | .19E-01 | .24E-06 | .60E-06 | .15E-02 | .64E-02 |
| 1E-3 | 0 | 0 | .23E-05 | .39E-05 | .23E-02 | .18E+00 | .24E-06 | .72E-06 | .69E-02 | .49E-01 |
| 5E-4 | 0 | 0 | .22E-05 | .32E-05 | .26E-02 | .11E+00 | .24E-06 | .83E-06 | .28E-01 | .85E-01 |
| 1E-4 | 0 | 3 | .26E-05 | .29E-05 | .27E-01 | .25E+00 | .95E-06 | .36E-06 | .15E+01 | .19E+00 |
| 5E-5 | 1 | 6 | .31E-05 | .32E-05 | .45E-02 | .71E+00 | .83E-06 | .42E-06 | .26E+01 | .92E+00 |
| 1E-5 | 3 | 8 | .29E-05 | .26E-05 | .22E-02 | .67E+00 | .72E-06 | .72E-06 | .31E+01 | .14E+01 |
| 5E-6 | 4 | 9 | .26E-05 | .27E-05 | .36E-01 | .54E+00 | .48E-06 | .72E-06 | .32E+01 | .21E+01 |
| 0.0 | 16 | 0 | .19E-05 | .26E-05 | .15E-05 | .50E-05 | .24E-06 | .60E-06 | .28E+01 | .40E+01 |

5 blocks of order 10; $n = 50$

| | udc deflations | | $\|H\hat{W} - \hat{W}\hat{\Lambda}\|$ | | $\|\hat{W}^*\hat{W} - I\|$ | | $\max_{1\leq j\leq n} \|\theta_j - \hat{\theta}_j\|$ | | $\|\,\|W\| - \|\hat{W}\|\,\|$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| $\epsilon$ | Type 1 | Type 2 | udc | comqr2 | udc | comqr2 | udc | comqr2 | udc | comqr2 |
| 1E-1 | 0 | 2 | .96E-05 | .91E-05 | .42E-04 | .34E-02 | .48E-06 | .13E-05 | .15E-02 | .16E-02 |
| 5E-2 | 0 | 2 | .17E-04 | .86E-05 | .11E-03 | .42E-02 | .48E-06 | .15E-05 | .11E-02 | .25E-02 |
| 1E-2 | 0 | 9 | .84E-05 | .73E-05 | .45E-03 | .23E-01 | .48E-06 | .77E-06 | .74E-02 | .15E-01 |
| 5E-3 | 2 | 13 | .85E-05 | .68E-05 | .62E-03 | .72E-01 | .48E-06 | .75E-06 | .43E-01 | .29E-01 |
| 1E-3 | 3 | 21 | .44E-05 | .87E-05 | .35E-02 | .79E+00 | .48E-06 | .97E-06 | .33E-01 | .33E+00 |
| 5E-4 | 3 | 21 | .40E-05 | .79E-05 | .69E-02 | .11E+01 | .48E-06 | .11E-05 | .53E-01 | .58E+00 |
| 1E-4 | 6 | 28 | .50E-05 | .81E-05 | .27E-01 | .29E+01 | .95E-06 | .17E-05 | .33E+01 | .20E+01 |
| 5E-5 | 20 | 31 | .62E-05 | .70E-05 | .22E-01 | .22E+01 | .13E-05 | .89E-06 | .53E+01 | .34E+01 |
| 1E-5 | 46 | 30 | .65E-05 | .72E-05 | .31E-01 | .13E+01 | .95E-06 | .12E-05 | .69E+01 | .53E+01 |
| 5E-6 | 61 | 24 | .69E-05 | .65E-05 | .17E-01 | .17E+01 | .72E-06 | .13E-05 | .76E+01 | .58E+01 |
| 0.0 | 98 | 0 | .37E-05 | .46E-05 | .27E-05 | .64E-05 | .48E-06 | .83E-06 | .85E+01 | .79E+01 |

Table 3: Nearly block diagonal unitary matrices

parameters are then given by $\gamma_{lp+j} = \gamma_j$, $\sigma_{lp+j} = \sigma_j$, $1 \leq j \leq p$, $1 \leq l < k$. Finally, we set $\gamma_n = 1$. If $\epsilon = 0$, then the eigenvalues of $H$ occur with multiplicity $k$. This experiment was performed for various values of $p$ and $k$ and for decreasing values of $\epsilon$.

The loss of accuracy and orthogonality in the eigenvectors as $\epsilon$ tends to zero is expected, since the eigenvectors are not well determined if $\epsilon = 0$. Nevertheless, the errors in eigenvalues and the eigenresiduals indicate that the algorithm is producing accurate results in this case.

Figure 1 shows the average CPU times required by single-precision versions of UDC, UDCP, COMQR2 and COMQR over five runs for $3 \leq n \leq 100$. This timing experiment was performed on a VAXstation 2000. We see that, although UDC and COMQR2 both require $O(n^3)$ arithmetic operations, UDC becomes substantially faster as $n$ increases. (We found that the time required for the explicit computation of the Hessenberg matrix $H$, which is needed for input to COMQR2 and COMQR, was negligible.) The crossover of the times required by UDCP and COMQR did not occur until $n = 51$. However, COMQR computes only the eigenvalues of $H$, and would have to be modified to provide the partial spectral resolution. Moreover, as $n$ becomes large, the computation of the full spectral resolution using UDC requires less CPU time than the computation of the eigenvalues only

| | $p = 15$, $k = 2$ | | | $p = 10$, $k = 3$ | | |
|---|---|---|---|---|---|---|
| | | CPU seconds | | | CPU seconds | |
| $\epsilon$ | udc deflations | udc | comqr2 | udc deflations | udc | comqr2 |
| 1.0e-01 | 0 | 0.114e+02 | 0.140e+02 | 0 | 0.117e+02 | 0.135e+02 |
| 1.0e-02 | 0 | 0.120e+02 | 0.138e+02 | 2 | 0.115e+02 | 0.129e+02 |
| 1.0e-03 | 3 | 0.115e+02 | 0.131e+02 | 9 | 0.102e+02 | 0.125e+02 |
| 1.0e-04 | 11 | 0.772e+01 | 0.130e+02 | 16 | 0.763e+01 | 0.122e+02 |
| 1.0e-05 | 16 | 0.597e+01 | 0.134e+02 | 28 | 0.470e+01 | 0.123e+02 |
| 1.0e-06 | 18 | 0.553e+01 | 0.131e+02 | 38 | 0.360e+01 | 0.130e+02 |
| 0.0 | 18 | 0.557e+01 | 0.891e+01 | 38 | 0.360e+01 | 0.700e+01 |

| | $p = 6$, $k = 5$ | | | $p = 5$, $k = 6$ | | |
|---|---|---|---|---|---|---|
| | | CPU seconds | | | CPU seconds | |
| $\epsilon$ | udc deflations | udc | comqr2 | udc deflations | udc | comqr2 |
| 1.0e-01 | 0 | 0.113e+02 | 0.134e+02 | 0 | 0.111e+02 | 0.133e+02 |
| 1.0e-02 | 4 | 0.111e+02 | 0.122e+02 | 2 | 0.121e+02 | 0.123e+02 |
| 1.0e-03 | 10 | 0.109e+02 | 0.121e+02 | 5 | 0.126e+02 | 0.115e+02 |
| 1.0e-04 | 21 | 0.785e+01 | 0.116e+02 | 17 | 0.953e+01 | 0.116e+02 |
| 1.0e-05 | 37 | 0.452e+01 | 0.104e+02 | 40 | 0.433e+01 | 0.110e+02 |
| 1.0e-06 | 54 | 0.230e+01 | 0.106e+02 | 61 | 0.185e+01 | 0.945e+01 |
| 0.0 | 54 | 0.232e+01 | 0.468e+01 | 61 | 0.183e+01 | 0.402e+01 |

| | $p = 3$, $k = 10$ | | | $p = 2$, $k = 15$ | | |
|---|---|---|---|---|---|---|
| | | CPU seconds | | | CPU seconds | |
| $\epsilon$ | udc deflations | udc | comqr2 | udc deflations | udc | comqr2 |
| 1.0e-01 | 0 | 0.104e+02 | 0.133e+02 | 34 | 0.780e+01 | 0.142e+02 |
| 1.0e-02 | 2 | 0.114e+02 | 0.122e+02 | 34 | 0.797e+01 | 0.129e+02 |
| 1.0e-03 | 10 | 0.117e+02 | 0.122e+02 | 56 | 0.497e+01 | 0.124e+02 |
| 1.0e-04 | 14 | 0.113e+02 | 0.120e+02 | 56 | 0.502e+01 | 0.123e+02 |
| 1.0e-05 | 42 | 0.607e+01 | 0.124e+02 | 60 | 0.470e+01 | 0.124e+02 |
| 1.0e-06 | 73 | 0.128e+01 | 0.106e+02 | 90 | 0.783e+00 | 0.131e+02 |
| 0. | 78 | 0.105e+01 | 0.315e+01 | 90 | 0.767e+00 | 0.227e+01 |

Table 4: Timings for problems with massive deflation

using COMQR.

Table 4 displays timings for UDC and COMQR2 for problems of order $n = 30$ generated as in Example 3 with $p$ and $k$ as indicated. We see that as $\epsilon$ tends to zero and the number of deflations performed by UDC increases, the computational time required by UDC decreases substantially as expected. In fact, the ratio of the time required by UDC when no deflations is performed to the time required when $\epsilon = 0$ (i.e., when the matrix is block diagonal with $k$ diagonal blocks) is roughly equal to $k$. Also note that the time required by COMQR2 to solve these problems generally does not change significantly until $\epsilon = 0$.

Thus, our experiments indicate that the subroutines UDC and UDCP are competitive with the subroutines COMQR2 and COMQR of EISPACK with respect to both accuracy and computing time.

**Acknowledgement**

We would like to thank Bill Gragg for discussions.

## References

[1] Ammar, G.S., Gragg, W.B., and Reichel, L., Determination of Pisarenko frequency estimates as eigenvalues of an orthogonal matrix. In *SPIE vol. 826, Advanced Algorithms and Architectures for Signal Processing II*, F. Luk, Ed. 1987, pp. 143–145.

[2] Arbenz, P., and Golub, G.H., On the spectral decomposition of Hermitian matrices modified by low rank perturbations with applications. *SIAM J. Matrix Anal. Appl. 9* (1988), 40–58.

[3] Cuppen, J.J.M., A divide and conquer method for the symmetric tridiagonal eigenproblem. *Numer. Math. 36* (1981), 177–195.

[4] Dongarra, J.J., and Sorensen, D.C., A fully parallel algorithm for the symmetric eigenvalue problem. *SIAM J. Sci. Stat. Comput. 8* (1987), s139–s154.

[5] Gragg, W.B., Positive definite Toeplitz matrices, the Arnoldi process for isometric operators and Gaussian quadrature on the unit circle (in Russian). In *Numerical Methods in Linear Algebra*, Nikolaev, E.S., Ed. Moscow University Press, Moscow, 1982, pp. 16–32.

[6] Gragg, W.B., and Reichel, L., A divide and conquer algorithm for the unitary eigenproblem. In *Hypercube Multiprocessors 1987*, Heath, M.T., Ed. SIAM, Philadelphia, 1987, pp. 639–647.

[7] Gragg, W.B., and Reichel, L., A divide and conquer method for unitary and orthogonal eigenproblems. *Numer. Math. 57* (1990), 695–718.

[8] Grenander, U., and Szegő, G., *Toeplitz Forms and Their Applications*. Chelsea, New York, 1984.

[9] Ipsen, I.C.F., and Jessup, E.R., Solving the symmetric tridiagonal eigenvalue problem on the hypercube. *SIAM J. Sci. Stat. Comput. 11* (1990), 203–229.

[10] Jones, W.B., Njåstad, O., and Thron, W.J., Moment theory, orthogonal polynomials, quadrature, and continued fractions associated with the unit circle. *Bull. London Math. Soc. 21* (1989), 113–152.

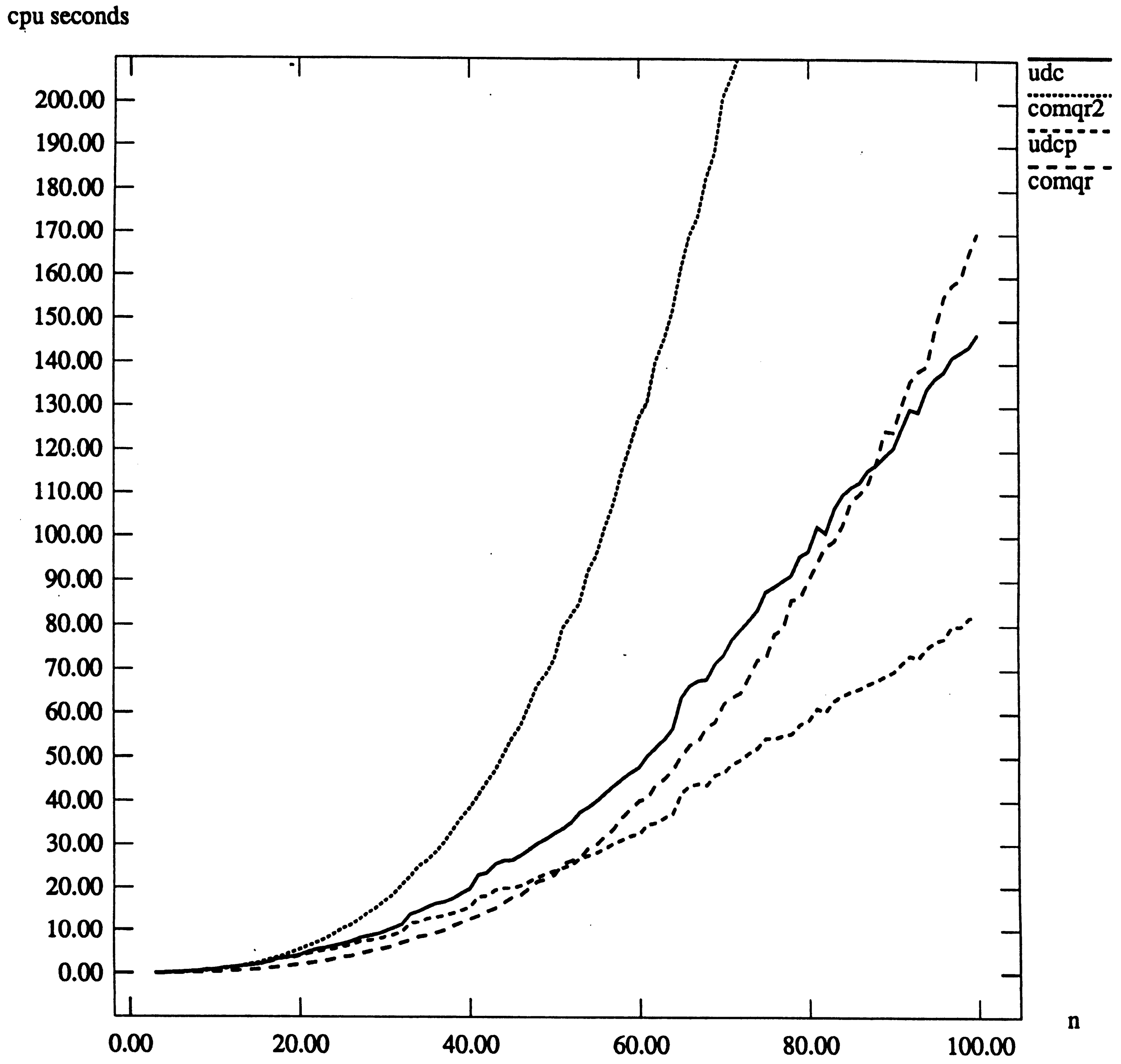
[11] Krishnakumar, A.S., and Morf, M., Eigenvalues of a symmetric tridiagonal matrix: a divide-and-conquer approach. *Numer. Math. 48* (1986), 349–368.

[12] Lawson, C., Hanson, R., Kincaid, D., and Krogh, F., Basic linear algebra subprograms for FORTRAN usage. *ACM Trans. Math. Software 5* (1979), 308–323.

[13] Reichel, L., and Ammar, G.S., Fast approximation of dominant harmonics by solving an orthogonal eigenvalue problem. In *Proceedings of the Second IMA Conference on Mathematics in Signal Processing*, J. McWhirter, Ed., Oxford University Press, to appear.

[14] Smith, B.T., Boyle, J.M., Ikebe, Y., Klema, V.C., and Moler, C.B., *Matrix Eigensystem Routines – EISPACK Guide*. 2nd ed., Springer, Berlin, 1970.

# Figure 1


cpu seconds
200.00
190.00
180.00
170.00
160.00
150.00
140.00
130.00
120.00
110.00
100.00
90.00
80.00
70.00
60.00
50.00
40.00
30.00
20.00
10.00
0.00
0.00
20.00
40.00
60.00
80.00
100.00
n
udc
comqr2
udcp
comqr