HEWLETT PACKARD

A Multishift QR Iteration without Computation of the Shifts

Augustin Dubrulle, Gene Golub*
Computer Systems Laboratory
HPL-93-54
July, 1993

matrix computations, eigenvalues, QR algorithm

Each iteration of the multishift QR algorithm of Bai and Demmel requires the computation of a "shift vector" defined by m shifts of the origin of the spectrum, which control the convergence of the process. A common choice of shifts consists of the eigenvalues of the trailing principal submatrix of order m, and current practice includes the computation of these eigenvalues in the determination of the shift vector. In this paper, we describe an algorithm based on the evaluation of the characteristic polynomial of a Hessenberg matrix, which directly produces the shift vector without computing eigenvalues. This algorithm is stable, more accurate, faster, and simpler than the current alternative. It also allows for a consistent shift strategy with dynamic adjustment of the number of shifts.

Internal Accession Date Only
* Stanford University, Palo Alto, California,
Presented at the *1993 Householder Conference*, UCLA Lake Arrowhead Center, California, June 14, 1993. Will also be released as a Stanford Computer Science Report.

A MULTISHIFT QR ITERATION WITHOUT COMPUTATION OF THE SHIFTS

AUGUSTIN A. DUBRULLE
Hewlett-Packard Laboratories
1501 Page Mill Road
Palo Alto, CA 94304
dubrulle@hpl.hp.com

GENE H. GOLUB[1]
Computer Science Department
Stanford University
Stanford, CA 94305
golub@sccm.stanford.edu

February 1, 1993

ABSTRACT

Each iteration of the multishift QR algorithm of Bai and Demmel requires the computation of a "shift vector" defined by m shifts of the origin of the spectrum, which control the convergence of the process. A common choice of shifts consists of the eigenvalues of the trailing principal submatrix of order m, and current practice includes the computation of these eigenvalues in the determination of the shift vector. In this paper, we describe an algorithm based on the evaluation of the characteristic polynomial of a Hessenberg matrix, which directly produces the shift vector without computing eigenvalues. This algorithm is stable, more accurate, faster, and simpler than the current alternative. It also allows for a consistent shift strategy with dynamic adjustment of the number of shifts.

1 Introduction

For the past thirty years, Francis' implicit QR algorithm for real Hessenberg matrices [6] has been at the core of the most effective software for the computation of the eigenvalues of real dense matrices. This remarkable longevity

[1]The work of this author was in part supported by the National Science Foundation, grant number ECS-9003107, and the Army Research Office, grant number DAA-03-91-G-0038.

has been recently extended by a multishift implementation of the method [1, 2] designed to exploit the architectural quirks of high-performance computers. Crucial to the fast convergence of the algorithm is the choice of certain parameters that shift the origin of the spectrum at each iteration. The shifts most commonly used – and most effective to date – are provided by the eigenvalues of a trailing principal submatrix of the iterated matrix. This strategy, conditionally recommended by Francis for his "double iteration," was made systematic by Wilkinson [18]. The implementation of the QR algorithm in its implicit form does not actually require that the shifts be known by value, but rather that a means to evaluate a polynomial with zeros equal to the shifts be available, and that this evaluation be feasible at matrix values of the variable. For the above shift strategy, this polynomial reduces to the characteristic polynomial of a Hessenberg matrix, and in the case of the double iteration, its coefficients are simply obtained as the trace and the determinant of a submatrix of order two. In the case of the multishift iteration, higher dimensions make conditions more complicated, and the implementation in [1] resorts to the explicit computation of the shifts for use in the evaluation of the polynomial. This is the part of the algorithm on which we focus here, based on the consideration that the evaluation of a polynomial should be of lesser complexity than the computation of the zeros. The scheme that we propose as a replacement is a matrix extension of Hyman's method that entirely avoids the computation of the shifts and offers much flexibility to adjust the number of shifts from one iteration to the next. The paper is organized as follows. Section 2 gives the necessary background on the implicit QR iteration, including the part of the computation relevant to the shifts. The derivation of our algorithm is presented in Section 3. Results of experiments are summarized in Section 4.

2 Background

Bai and Demmel [2] recently developed a multishift version of Francis' implicit QR algorithm [6] that has proved to be quite efficient with high-performance work-stations and vector computers. For appropriate values of the shift multiplicity, the algorithm has good convergence properties and lends itself to formulations in terms of matrix-vector [1] and matrix-matrix operations [5] that can exploit machine architecture. It is based on those same properties of the QR algorithm from which Francis devised the implicit double iteration. Skipping details that can be found in the above references, we give below an outline of the method and its foundations.

For $\mathbf{A} \in \mathrm{C}^{n\times n}$ and $\mathbf{A}^{(1)} = \mathbf{A}$, the QR iteration with explicit shift is

represented by the equations

$$\mathbf{A}^{(k)} - \sigma_k \mathbf{I} = \mathbf{Q}^{(k)}\mathbf{R}^{(k)}, \qquad \mathbf{A}^{(k+1)} = \sigma_k \mathbf{I} + \mathbf{R}^{(k)}\mathbf{Q}^{(k)}, \tag{2.1}$$

where σ_k is some shift of the origin of the spectrum of $\mathbf{A}^{(k)}$, and the matrices $\mathbf{Q}^{(k)}$ and $\mathbf{R}^{(k)}$ are defined as the unitary and upper-triangular factors of a QR decomposition [8]. The iterates are mutually similar,

$$\mathbf{A}^{(k+1)} = \mathbf{Q}^{(k)*}\mathbf{A}^{(k)}\mathbf{Q}^{(k)},$$

and converge to triangular form,

$$\lim_{k\to\infty} |a_{ij}^{(k)}| = \lim_{k\to\infty} |r_{ij}^{(k)}|,$$

to reveal the eigenvalues of $\mathbf{A}$. Shifts close to an eigenvalue accelerate convergence to that eigenvalue. With simple choices of shifts, earliest convergence usually manifest itself in the south-east corner of the iterated matrix, which can be deflated by its last row and column as soon as an eigenvalue is found (the off-diagonal elements of the last row are negligible). The algorithm has three important properties:

- the QR iteration preserves the Hessenberg form;
- the unitary matrix $\mathbf{\Omega}^{(k)}$ that defines the $(k+1)^{st}$ iterate by

$$\mathbf{A}^{(k+1)} = \mathbf{\Omega}^{(k)*}\mathbf{A}^{(1)}\mathbf{\Omega}^{(k)}, \qquad \mathbf{\Omega}^{(k)} = \mathbf{Q}^{(1)}\mathbf{Q}^{(2)}\dots\mathbf{Q}^{(k)},$$

  derives from the QR decomposition

$$(\mathbf{A}^{(1)} - \sigma_1\mathbf{I})(\mathbf{A}^{(1)} - \sigma_2\mathbf{I})\dots(\mathbf{A}^{(1)} - \sigma_k\mathbf{I}) = \mathbf{\Omega}^{(k)}\mathbf{U}^{(k)},$$

$$\mathbf{U}^{(k)} = \mathbf{R}^{(1)}\mathbf{R}^{(2)}\dots\mathbf{R}^{(k)};$$

- if $\mathbf{A}^{(1)}$ is an unreduced Hessenberg matrix, $\mathbf{\Omega}^{(k)}$ is entirely determined by the vector

$$(\mathbf{A}^{(1)} - \sigma_1\mathbf{I})(\mathbf{A}^{(1)} - \sigma_2\mathbf{I})\dots(\mathbf{A}^{(1)} - \sigma_k\mathbf{I})\mathbf{e}_1.$$

Based on these properties and the uniqueness of the QR decomposition, Francis devised an iteration for unreduced real upper-Hessenberg matrices that avoids complex iterates and converges to a real matrix departing from triangularity only by some blocks of order two on the diagonal. This "double iteration", which is equivalent to two single iterations (2.1) with the same

shifts, is described by the following equations where $\mathbf{H} \in \mathrm{R}^{n\times n}$ is upper-Hessenberg and $\mathbf{H}^{(1)} = \mathbf{H}$:

$$\begin{aligned}
&(a)\quad \mathbf{z}_2^{(k)} = (\mathbf{H}^{(k)} - \sigma_k\mathbf{I})(\mathbf{H}^{(k)} - \sigma_{k+1}\mathbf{I})\mathbf{e}_1,\\
&\qquad \mathbf{S}^{(k+2)*}\mathbf{z}_2^{(k)} = \pm\|\mathbf{z}_2^{(k)}\|_2\mathbf{e}_1,\\
&(b)\quad \mathbf{B}^{(k+2)} = \mathbf{S}^{(k+2)*}\mathbf{H}^{(k)}\mathbf{S}^{(k+2)},\\
&(c)\quad \mathbf{H}^{(k+2)} = \mathbf{P}^{(k+2)*}\mathbf{B}^{(k+2)}\mathbf{P}^{(k+2)}.
\end{aligned} \tag{2.2}$$

Equations (2.2.a) define the "shift transformation" represented by the Householder matrix $\mathbf{S}^{(k+2)}$ that reduces the "shift vector" $\mathbf{z}_2^{(k)}$ to a stretching of $\mathbf{e}_1$ with a sign determined by some stability condition. A choice of real or complex-conjugate values for the shifts σ_k and σ_{k+1} ensures that all elements of the computation remain real. The Hessenberg form is lost for $\mathbf{B}^{(k+2)}$ in the shift transformation (2.2.b), but it is restored for $\mathbf{H}^{(k+2)}$ by an appropriate sequence of Householder transformations represented by the orthogonal matrix $\mathbf{P}^{(k+2)}$ of equation (2.2.c). The formal relationship between this iteration and iteration (2.1) for $\mathbf{A} = \mathbf{H}$ is given by

$$\mathbf{S}^{(k+2)}\mathbf{P}^{(k+2)} = \mathbf{Q}^{(k)}\mathbf{Q}^{(k+1)}.$$

Note that here, in contrast to algorithm (2.1), the shifts are "implicitly" implanted in the process through $\mathbf{z}_2^{(k)}$ and $\mathbf{S}^{(k+2)}$, and that they do not have to be known by value since their sum and product are enough to define $\mathbf{z}_2^{(k)}$. The most popular implementations of the algorithm use for shifts the eigenvalues of the trailing principal submatrix of order two of $\mathbf{H}^{(k)}$, and the shift vector is derived from the trace and the determinant of that submatrix (see, for example, subroutine *hqr* of EISPACK [14]). Convergence of the iteration generally results in the separation of trailing submatrices of orders one or two, for which immediate deflation is effected simply by dropping the corresponding rows and columns.

Bai and Demmel generalized this scheme to the m-tuple iteration ($m \geq 2$) represented by the following equations,

$$\begin{aligned}
&(a)\quad \mathbf{z}_m^{(k)} = \prod_{i=1}^{m}(\mathbf{H}^{(k)} - \sigma_{k+i-1}\mathbf{I})\mathbf{e}_1,\\
&\qquad \mathbf{S}^{(k+m)*}\mathbf{z}_m^{(k)} = \pm\|\mathbf{z}_m^{(k)}\|_2\mathbf{e}_1,\\
&(b)\quad \mathbf{B}^{(k+m)} = \mathbf{S}^{(k+m)*}\mathbf{H}^{(k)}\mathbf{S}^{(k+m)},\\
&(c)\quad \mathbf{H}^{(k+m)} = \mathbf{P}^{(k+m)*}\mathbf{B}^{(k+m)}\mathbf{P}^{(k+m)},
\end{aligned} \tag{2.3}$$

where the transformation matrices are orthogonal, and equation (2.3.c) represents a reduction to Hessenberg form. The shifts are the eigenvalues of $\mathbf{T}^{(k)}$, the trailing principal submatrix of order m of $\mathbf{H}^{(k)}$ (this choice again keeps the computation real). Convergence [16] manifests itself by the separation of a trailing principal submatrix of order often close to m, which provides for automatic deflation. In general, for moderate values of m, the algorithm has good convergence properties and allows for the efficient use of high-performance computers. In [2] and [1], the shift vector is computed as formally prescribed by the first of equations (2.3.a), that is, from the eigenvalues of $\mathbf{T}^{(k)}$. This is the part of the algorithm on which we shall focus, based on the following observations. First, the above shift strategy defines the first of equations (2.3.a) as

$$\mathbf{z}_m^{(k)} = p_m^{(k)}(\mathbf{H}^{(k)})\mathbf{e}_1, \qquad p_m^{(k)}(\mathbf{T}^{(k)}) = \mathbf{0}, \tag{2.4}$$

where $p_m^{(k)}(\mu)$ is the characteristic polynomial of $\mathbf{T}^{(k)}$. Since the calculation of the shift vector essentially amounts to the evaluation of the characteristic polynomial of a Hessenberg matrix, it seems unnecessarily complicated to perform this evaluation by computing eigenvalues. Second, it has been observed that the accuracy of the shift vector is sensitive to the order in which the shifts enter the computation. While the incorporation of some sorting procedure in the implementation of the algorithm can solve this problem, it would certainly be preferable to use a scheme to which the ordering of the shifts is irrelevant. Such questions are resolved by the method that we propose in the next section.

3 The direct evaluation of the shift vector

From definition (2.2), the shift transformation is invariant under scaling of $\mathbf{z}_m^{(k)}$, and $p_m^{(k)}(\mu)$ needs to be defined only to a multiplicative constant. Accordingly, we shall use the term "characteristic polynomial" and the notation $p_m^{(k)}(\mu)$ to designate any scaled value of the characteristic polynomial. For simplicity, we shall also omit iteration superscripts in equations.

The determination of the characteristic polynomial of a matrix was a major topic of early numerical analysis in connection with the solution of the general eigenvalue problem. Up until the end of the 1930s, much effort was devoted to the computation of the coefficients as a preliminary step to the computation of the zeros. The best-known methods devised for that purpose are due to Le Verrier [15], Krylov [10], and Danilevskii [3]. Lanczos [11] later developed an algorithm of similarity tridiagonalization that reduces the evaluation of the polynomial to the implementation of a simple

three-term recurrence. All these schemes are costly of computation and have been shown to suffer from numerical instability [17]. They were replaced in the 1950s by algorithms specifically designed for the direct evaluation of the characteristic polynomial of Hessenberg matrices [4, 9, 17]. Hyman's method, which constitutes the basis for our work, is the best of the latter for its simplicity, efficiency, stability, and accuracy. It was effectively used by Parlett [13] with Laguerre's iteration to produce eigenvalue solvers outperformed only by their QR competitors. While it is generally thought of as a scalar algorithm, its extension to the vector form (2.4) is straight-forward, as demonstrated below.

We first summarize the original (scalar) form of the method. For a real, unreduced, upper-Hessenberg matrix $\mathbf{T}$ of order m and a scalar μ, consider the system of linear equations

$$(\mu\mathbf{I} - \mathbf{T})\mathbf{x} = p_m\mathbf{e}_1 \tag{3.1}$$

to be solved for $p_m = p_m(\mu)$, given $x_m = 1$. Cramer's rule applied to x_m yields the identity

$$1 = (-1)^{m-1}\frac{D_{1m}}{\Delta_m}p_m,$$

where D_{1m} is the adjugate minor of t_{1m}, and $\Delta_m = \Delta_m(\mu)$ is the determinant of the system, that is, the value of the characteristic polynomial of $\mathbf{T}$ at μ. Since $\mathbf{T}$ is upper-Hessenberg, D_{1m} is the determinant of a triangular matrix the main diagonal of which is the sub-diagonal of $\mathbf{T}$:

$$D_{1m} = \prod_{i=1}^{m-1} t_{i+1i}.$$

Hence, p_m differs from Δ_m only by a multiplicative constant, and therefore satisfies the requirements of our problem. The numerical solution of the system (3.1) for p_m results from using in the first equation the values of $x_{m-1}, x_{m-2}, \ldots, x_1$ obtained from the last $m-1$ equations by backward substitution. This process can be described by the following recurrence,

$$\begin{aligned} x_m &= 1, \qquad p_1 = \mu - t_{mm}, \\ x_{m-i} &= t_{m-i+1\,m-i}^{-1}\,p_i, \\ p_{i+1} &= \mu x_{m-i} - \sum_{j=m-i}^{m} t_{m-i\,j}x_j, \end{aligned} \qquad i = 1, 2, \ldots, m-1, \tag{3.2}$$

which provides the values at μ of all the trailing principal minors p_k, $k < m$, of $\mu\mathbf{I} - \mathbf{T}$. As the occurrence of small sub-diagonal elements may cause

overflow of floating-point representation, the implementation of the method requires some monitoring of the magnitude of x_i and, possibly, scaling. If $|\mu|$ is bounded by some norm of $\mathbf{T}$, accurate results can be expected [17].

We now derive the matrix algorithm to compute the shift vector of degree m,

$$\mathbf{z}_m = p_m(\mathbf{H})\mathbf{e}_1,$$

for some Hessenberg matrix $\mathbf{H}$ of order n, $n > m$, whose trailing principal submatrix of order m may coincide with $\mathbf{T}$. Equations (3.2) are easily promoted to the matrix level by substituting $\mathbf{T}$ for μ in the expressions of $p_j = p_j(\mu)$ and $x_j = x_j(\mu)$. Then, defining

$$\mathbf{w}_j = x_j(\mathbf{H})\mathbf{e}_1,$$

we immediately get the recurrence

$$\begin{aligned}
\mathbf{w}_m &= \mathbf{e}_1, \qquad \mathbf{z}_1 = \mathbf{h}_1 - t_{mm}\mathbf{e}_1, \\
\mathbf{w}_{m-i} &= t_{m-i+1\,m-i}^{-1}\,\mathbf{z}_i, \qquad i = 1,2,\ldots,m-1, \qquad (3.3) \\
\mathbf{z}_{i+1} &= \mathbf{H}\mathbf{w}_{m-i} - \sum_{j=m-i}^{m} t_{m-i\,j}\mathbf{w}_j,
\end{aligned}$$

which provides the shift vectors of all degrees up to m. Note that $\mathbf{w}_m, \mathbf{w}_{m-1}, \ldots, \mathbf{w}_i$ form a triangular matrix, and constitute a basis for the subspace of dimension $m-i+1$ if $\mathbf{H}$ is unreduced. Note also that each $\mathbf{w}_i$ is a shift vector for p_{m-i} since it is a scaled copy of $\mathbf{z}_{m-i}$. Hence, by the time any $\mathbf{z}_j$ is computed, the choice to adjust the number of shifts to a smaller value is available at no cost and without violation of the shift strategy if the set $\{\mathbf{w}_i \mid i < j\}$ is kept in memory (a natural way to organize the computation). This is in sharp contrast with the approach based on the calculation of the shifts, where adherence to the strategy does not permit the construction of shift vectors other than $\mathbf{z}_m$ for a given value of m. Finally, the amount of computation required for Hyman's method is of the order of $2m^3/3$ operations, which compares quite favorably with an order of $10m^3 + m^3/3$ operations for the eigenvalues of $\mathbf{T}$ (average) and the construction of the shift vector proper. In the context of the QR algorithm, the gains obtained in practice will be somewhat lower than the ratio of these counts, mostly because efficiency considerations require that m take modest values.

While the computation (3.2) is essentially a backward substitution performed row-wise, it can also be organized as a forward substitution proceeding column-wise by solving the system

$$\mathbf{x}^T(\mu\mathbf{I} - \mathbf{T}) = p_m\mathbf{e}_m^T$$

for $p_m = p_m(\mu)$, given $x_1 = 1$. The associated vector algorithm is described by the following recurrence:

$$\begin{aligned} \mathbf{w}_1 &= \mathbf{e}_1, \qquad \mathbf{z}_1 = \mathbf{h}_1 - t_{11}\mathbf{e}_1, \\ \mathbf{w}_i &= t_{i\,i-1}^{-1}\,\mathbf{z}_{i-1}, \qquad\qquad i = 2, \ldots, m. \\ \mathbf{z}_i &= \mathbf{H}\mathbf{w}_i - \sum_{j=1}^{i} t_{ji}\mathbf{w}_j, \end{aligned} \tag{3.4}$$

In this scheme, $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_i$ form a triangular matrix of order i, and $\mathbf{z}_i$ (or $\mathbf{w}_{i+1}$) is the shift vector associated with the roots of $\mathbf{T}$'s leading principal submatrix of order i. Because of this feature, algorithms (3.4) and (3.3) are not interchangeable for the adjustment of the number of shifts in the course of the computation.

4 Experiments

Our experiments were conducted on an HP 9000/720 work-station (IEEE double precision), using two types of random matrices for $\mathbf{H}$. The first type is obtained by orthogonal reduction to Hessenberg form of matrices with elements uniformly distributed in $(-1, +1)$. The second type, which consists of Hessenberg matrices with elements uniformly distributed in $(-1, +1)$, is known for causing convergence difficulties in the QR iteration when m takes moderately large values [5].

In a first set of runs for $100 \leq n \leq 300$ and $m \leq n/2$, our algorithm was eight to ten times as fast as its competitor for the computation of shift vectors, both implementations incorporating the same monitoring and threshold-scaling procedures[2] for intermediate results. In other experiments involving two levels of machine precision, we verified that, as expected, the accuracy delivered by our algorithm is very good, and better than that which could be obtained from the computation of eigenvalues. The relative error measured in the $\ell_1(m)$ norm for the shift vector was consistently bounded by a value near $2m\varepsilon$, for a machine precision ε. To achieve comparable results with the other approach requires that the shifts be entered in the computation in some well-defined order (the order of decreasing modulus gave good accuracy with our matrices). For relatively large values of m ($m \approx 100$) we

[2]For economy of computation, we try to scale as infrequently as possible, but the norm of each $\mathbf{w}_i$, $1 \leq i \leq m$, must be computed and tested. Much of this work could be avoided with the availability of an indicator of floating-point overflow that could be tested with very little performance penalty.

encountered cases where the order of the shifts returned by *hqr* caused the accuracy of the shift vector to drop to only a few digits.

As expected, the impact of the new algorithm on the time performance of a QR eigenvalue solver was found to be less than dramatic, as modest optimum values of m make the QR iteration proper the largely dominant computation. We could measure gains of five to ten percent in the computation of eigenvalues alone for matrix orders up to 400 at near-optimum values of m. We did not, however, obtain measurements for cases where inaccuracies in the shift vector would seriously affect convergence.

5 Conclusion

For its simplicity, efficiency, and accuracy, our implementation of the shift strategy in the implicit QR algorithm is in all respects preferable to the current practice of computing the eigenvalues of a submatrix. If no dynamic ajustment of the number of shifts is considered, a FORTRAN implementation for large matrices should be based on formulation (3.4), which proceeds column-wise. Formulation (3.3) should be preferred for other programming languages, which have a contrary memory layout for two-dimensional arrays.

The algorithm's amenability to dynamic adjustment of the number of shifts remains, however, a good feature in search of an application, and could not be used to improve the behavior of the QR iteration for moderately large numbers m of shifts (roughly, $m \geq 20$). In such cases, the most efficient implementation of the QR iteration is likely to consist of a sequence of m/k multishift iterations, each using k of the shifts, $k \leq 16$. In experiments with $k = 2$ and $m = 30$, this approach was shown to be competitive with a block implementation of the m-shift iteration [5]. Unfortunately, it cannot use our algorithm and requires the computation of the shifts.

Finally, it must be noted that our scheme extends to the generalized eigenvalue problem, and simplifies the implementation of a multishift version of the QZ algorithm [12].

References

[1] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen, *LAPACK Users' Guide*, SIAM, Philadelphia PA, 1992.

[2] Z. Bai and J. W. Demmel, *On a block implementation of the Hessenberg multishift QR iteration*, Int. J. High-Speed Comp., 1(1):97-112, 1989.

[3] A. Danilevskii, *On the numerical solution of the secular equation*, Math. Sb., 2(54):169-171, 1937.

[4] L. Derwidué, *Une méthode mécanique de calcul des vecteurs propres d' une matrice quelconque*, Bull. Soc. Roy. Sci. Liège, 149-171, 1955.

[5] A. A. Dubrulle, *The multishift QR algorithm – is it worth the trouble?*, TR G320-3558, IBM Scientific Center, Palo Alto CA, 1991 (revised 1992).

[6] G. J. F. Francis, *The QR transformation, Parts I and II*, Comp. J., 4:265-272 and 4:332-345, 1961-1962.

[7] J. N. Franklin, *Matrix Theory*, Prentice-Hall, Englewood Cliffs NJ, 1968.

[8] G. H. Golub and C. F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore MD, 1989.

[9] M. A. Hyman, *Eigenvalues and eigenvectors of general matrices*, Twelfth ACM National Meeting, Houston TX, 1957.

[10] A. Krylov, *On the numerical solution of equations by which the frequency of small oscillations is determined in material systems*, Izv. Akad. Nauk SSSR Ser. Fiz.-Mat., 4:491-539 1931.

[11] C. Lanczos, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Res. Nat. Bur. Stand., 45: 255-282, 1950.

[12] C. B. Moler and G. W. Stewart, *An algorithm for generalized matrix eigenvalue problems*, SIAM. J. Num. Anal., 10:241-256, 1973.

[13] B. N. Parlett, *Laguerre's method applied to the matrix eigenvalue problem*, Math. Comp., 18:464-485, 1964.

[14] B. T. Smith, J. M. Boyle, Y. Ikebe, V. C. Klemma, and C. B. Moler, *Matrix Eigensystem Routines – EISPACK Guide*, Springer-Verlag, New York, NY, 1976.

[15] U. Le Verrier, *Sur les variations séculaires des éléments elliptiques des sept planètes principales*, J. Math. Pures et Appl., 5:220-254, 1840.

[16] D. S. WATKINS AND L. ELSNER, *Convergence of algorithms of decomposition type for the eigenvalue problem*, Lin. Alg. and Appl., 143:19-47, 1991.

[17] J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, 1965.

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