# Francis's Algorithm

**David S. Watkins**

**Abstract.** John Francis's implicitly shifted *QR* algorithm turned the problem of matrix eigenvalue computation from difficult to routine almost overnight about fifty years ago. It was named one of the top ten algorithms of the twentieth century by Dongarra and Sullivan, and it deserves to be more widely known and understood by the general mathematical community. This article provides an efficient introduction to Francis's algorithm that follows a novel path. Efficiency is gained by omitting the traditional but wholly unnecessary detour through the basic *QR* algorithm. A brief history of the algorithm is also included. It was not a one-man show; some other important names are Rutishauser, Wilkinson, and Kublanovskaya. Francis was never a specialist in matrix computations. He was employed in the early computer industry, spent some time on the problem of eigenvalue computation and did amazing work, and then moved on to other things. He never looked back, and he remained unaware of the huge impact of his work until many years later.

**1. INTRODUCTION.** A problem that arises frequently in scientific computing applications is that of computing the eigenvalues and eigenvectors of a real or complex square matrix. Nowadays we take it for granted that we can do this. For example, I was able to compute a complete set of eigenvalues and eigenvectors of a real $1000 \times 1000$ matrix on my laptop in about 15 seconds using MATLAB®. Not only MATLAB has this capability; it is built into a large number of packages, both proprietary and open source. Not long ago [**7**] the complete eigensystem of a dense matrix of order $10^5$ was computed on a parallel supercomputer.[1] These feats are made possible by good software *and* good hardware. Today's computers are fast and have large memories, and they are reliable.

Fifty years ago the situation was very different. Computers were slow, had small memories, and were unreliable. The software situation was equally bad. Nobody knew how to compute the eigenvalues of, say, a $10 \times 10$ matrix in an efficient, reliable way. It was in this environment that young John Francis found himself writing programs for the Pegasus computer at the National Defense Research Corporation in London. A major problem at that time was the flutter of aircraft wings, and for the analysis there was a need to compute eigenvalues. Francis was given the task of writing some matrix computation routines, including some eigenvalue programs. His interest in the problem grew far beyond what he had been assigned to do. He had obtained a copy of Heinz Rutishauser's paper on the *LR* algorithm [**13**], he had attended seminar talks by James Wilkinson[2] in which he learned about the advantages of computing with orthogonal matrices, and he saw the possibility of creating a superior method. He worked on this project on the side, with the tolerance (and perhaps support) of his supervisor, Christopher Strachey. In 1959 he created and tested the procedure that is now commonly known as the implicitly-shifted *QR* algorithm, which I prefer to call *Francis's*

doi:10.4169/amer.math.monthly.118.05.387

[1] Most large matrices that arise in applications are sparse, not dense. That is, the vast majority of their entries are zeros. For sparse matrices there are special methods [**1**] that can compute selected eigenvalues of extremely large matrices (much larger than $10^5$). These methods are important, but they are not the focus of this paper.

[2] A few years later, Wilkinson wrote the highly influential book *The Algebraic Eigenvalue Problem* [**20**].

*algorithm*. It became and has continued to be the big workhorse of eigensystem computations. A version of Francis's algorithm was used by MATLAB when I asked it to compute the eigensystem of a $1000 \times 1000$ matrix on my laptop. Another version was used for the computation on the matrix of order $10^5$ reported in [**7**].

Once Francis finished his work on the *QR* project, he moved on to other things and never looked back. In the infant computer industry there were many urgent tasks, for example, compiler development. Within a few years the importance of Francis's algorithm was recognized, but by then Francis had left the world of numerical analysis. Only many years later did he find out what a huge impact his algorithm had had.

**2. PRECURSORS OF FRANCIS'S ALGORITHM.** Most of us learned in an introductory linear algebra course that the way to compute eigenvalues is to form the characteristic polynomial and factor it to get the zeros. Because of the equivalence of the eigenvalue problem with that of finding the roots of a polynomial equation, it is difficult to put a date on the beginning of the history of eigenvalue computations. The polynomial problem is quite old, and we all know something about its history. In particular, early in the 19th century Abel proved that there is no general formula (using only addition, subtraction, multiplication, division, and the extraction of roots) for the roots of a polynomial of degree five. This result, which is one of the crowning achievements of Galois theory, has the practical consequence for numerical analysts that all methods for computing eigenvalues are iterative. Direct methods, such as Gaussian elimination for solving a linear system $Ax = b$, do not exist for the eigenvalue problem.

The early history of eigenvalue computation is intertwined with that of computing roots of polynomial equations. Eventually it was realized that forming the characteristic polynomial might not be a good idea, as the zeros of a polynomial are often extremely sensitive to small perturbations in the coefficients [**19, 21**]. All modern methods for computing eigenvalues work directly with the matrix, avoiding the characteristic polynomial altogether. Moreover, the eigenvalue problem has become a much more important part of computational mathematics than the polynomial root finding problem is.

For our story, an important early contribution was the 1892 dissertation of Jacques Hadamard [**9**], in which he proposed a method for computing the poles of a meromorphic function

$$f(z) = \sum_{k=0}^{\infty} \frac{s_k}{z^{k+1}}, \tag{1}$$

given the sequence of coefficients $(s_k)$. Some sixty years later Eduard Stiefel, founder of the Institute for Applied Mathematics at ETH Zürich, set a related problem for his young assistant, Heinz Rutishauser: given a matrix $A$, determine its eigenvalues from the sequence of moments

$$s_k = y^T A^k x, \quad k = 0, 1, 2, \ldots,$$

where $x$ and $y$ are (more or less) arbitrary vectors. If the moments are used to define a function $f$ as in (1), every pole of $f$ is an eigenvalue of $A$. Moreover, for almost any choice of vectors $x$ and $y$, the complete set of poles of $f$ is exactly the complete set of eigenvalues of $A$. Thus the problem posed by Stiefel is essentially equivalent to the problem addressed by Hadamard. Rutishauser came up with a new method, which he called the quotient-difference (qd) algorithm [**11, 12**], that solves this problem. Today

we know that this is a bad way to compute eigenvalues, as the poles of $f$ (like the zeros of a polynomial) can be extremely sensitive to small changes in the coefficients. However, Rutishauser saw that there were multiple ways to organize and interpret his qd algorithm. He reformulated it as a process of factorization and recombination of tridiagonal matrices, and from there he was able to generalize it to obtain the *LR* algorithm [**13**]. For more information about this seminal work see [**8**], in addition to the works of Rutishauser cited above.

Francis knew about Rutishauser's work, and so did Vera Kublanovskaya, working in Leningrad. Francis [**3**] and Kublanovskaya [**10**] independently proposed the *QR* algorithm, which promises greater stability by replacing *LR* decompositions with *QR* decompositions. The basic *QR* algorithm is as follows: Factor your matrix $A$ into a product $A = QR$, where $Q$ is unitary and $R$ is upper triangular. Then reverse the factors and multiply them back together to get a new matrix: $RQ = \hat{A}$. Briefly we have

$$A = QR, \qquad RQ = \hat{A}.$$

This is one iteration of the basic *QR* algorithm. It is easy to check that $\hat{A} = Q^{-1}AQ$, so $\hat{A}$ is unitarily similar to $A$ and therefore has the same eigenvalues. If the process is now iterated:

$$A_{j-1} = Q_j R_j, \qquad R_j Q_j = A_j,$$

the sequence of unitarily similar matrices so produced will (usually) tend toward (block) triangular form, eventually revealing the eigenvalues on the main diagonal. If $A$ is real, the entire process can be done in real arithmetic.

The basic *QR* algorithm converges rather slowly. A remedy is to incorporate shifts of origin:

$$A_{j-1} - \rho_j I = Q_j R_j, \qquad R_j Q_j + \rho_j I = A_j. \tag{2}$$

Again it is easy to show that the matrices so produced are unitarily similar. Good choices of shifts $\rho_j$ can improve the convergence rate dramatically. A good shift is one that approximates an eigenvalue well.

Many applications feature real matrices that have, however, lots of complex eigenvalues. If one wants rapid convergence of complex eigenvalues, one needs to use complex shifts and carry out (2) in complex arithmetic. This posed a problem for Francis: Working in complex arithmetic on the Pegasus computer would have been really difficult. Moreover, complex matrices require twice as much storage space as real matrices do, and Pegasus didn't have much storage space. He therefore looked for a method that avoids complex arithmetic.

If iterate $A_0$ is real and we do an iteration of (2) with a complex shift $\rho$, the resulting matrix $A_1$ is, of course, complex. However, if we then do another iteration using the complex conjugate shift $\overline{\rho}$, the resulting matrix $A_2$ is again real. Francis knew this, and he looked for a method that would get him directly from $A_0$ to $A_2$. He found such a method, the double-shift *QR* algorithm [**4**], which does two iterations of the *QR* algorithm implicitly, entirely in real arithmetic. This algorithm does not cause convergence to triangular form; it is impossible for complex eigenvalues to emerge on the main diagonal of a real matrix. Instead pairs of complex conjugate eigenvalues emerge in $2 \times 2$ packets along the main diagonal of a block triangular matrix.

Francis coded his new algorithm in assembly language on the Pegasus computer and tried it out. Using this new method, the flying horse was able to compute all of the eigenvalues of a $24 \times 24$ matrix in just over ten minutes [**4**].

We will describe Francis's double-shift $QR$ algorithm below, and the reader will see that it is completely different from the shifted $QR$ algorithm (2). Francis had to expend some effort to demonstrate that his new method does indeed effect two steps of (2).

When we teach students about Francis's algorithm today, we still follow this path. We introduce the shifted $QR$ algorithm (2) and talk about why it works, then we introduce Francis's algorithm, and then we demonstrate equivalence using the so-called implicit-$Q$ theorem. A primary objective of this paper is to demonstrate that this approach is inefficient. It is simpler to introduce Francis's algorithm straightaway and explain why it works, bypassing (2) altogether.[3]

**3. REFLECTORS AND THE REDUCTION TO HESSENBERG FORM.** Before we can describe Francis's algorithm, we need to introduce some basic tools of numerical linear algebra. For simplicity we will restrict our attention to the real case, but everything we do here can easily be extended to the complex setting.

Let $x$ and $y$ be two distinct vectors in $\mathbb{R}^n$ with the same Euclidean norm: $\|x\|_2 = \|y\|_2$, and let $\mathcal{S}$ denote the hyperplane (through the origin) orthogonal to $x - y$. Then the linear transformation $Q$ that reflects vectors through $\mathcal{S}$ clearly maps $x$ to $y$ and vice versa. Since $Q$ preserves norms, it is unitary: $Q^* = Q^{-1}$. It is also clearly involutory: $Q^{-1} = Q$, so it is also Hermitian: $Q^* = Q$. Viewing $Q$ as a matrix, it is real, orthogonal ($Q^T = Q^{-1}$), and symmetric ($Q^T = Q$). A precise expression for the matrix $Q$ is

$$Q = I - 2uu^T, \quad \text{where } u = (x - y)/\|x - y\|_2.$$

Now consider an arbitrary nonzero $x$ and a special $y$:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \qquad y = \begin{bmatrix} y_1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \qquad \text{where } y_1 = \pm\|x\|_2.$$

Since $x$ and $y$ have the same norm, we know that there is a reflector $Q$ such that $Qx = y$. Such an operator is clearly a boon to numerical linear algebraists, as it gives us a means of introducing large numbers of zeros. Letting $e_1$ denote the vector with a 1 in the first position and zeros elsewhere, as usual, we can restate our result as follows.

**Theorem 1.** *Let $x \in \mathbb{R}^n$ be any nonzero vector. Then there is a reflector $Q$ such that $Qx = \alpha e_1$, where $\alpha = \pm\|x\|_2$.*

In the world of matrix computations, reflectors are also known as *Householder transformations*. For details on the practical construction and use of reflectors see [**17**] or [**6**], for example.

With reflectors in hand we can easily transform any matrix nearly all the way to triangular form. A matrix $A$ is called *upper Hessenberg* if it satisfies $a_{ij} = 0$ whenever $i > j + 1$. For example, a $5 \times 5$ upper Hessenberg matrix has the form

[3]It is only fair to admit that this is not my first attempt. The earlier work [**16**], which I now view as premature, failed to expose fully the role of Krylov subspaces (explained below) for the functioning of Francis's algorithm. It is my fervent hope that I will not view the current work as premature a few years from now.

$$\begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ & * & * & * & * \\ & & * & * & * \\ & & & * & * \end{bmatrix},$$

where blank spots denote zeros and asterisks denote numbers that can have any real value.

**Theorem 2.** *Every $A \in \mathbb{R}^{n\times n}$ is orthogonally similar to an upper Hessenberg matrix: $H = Q^{-1}AQ$, where $Q$ is a product of $n-2$ reflectors.*

*Proof.* The first reflector $Q_1$ has the form

$$Q_1 = \begin{bmatrix} 1 & \\ & \tilde{Q}_1 \end{bmatrix}, \tag{3}$$

where $\tilde{Q}_1$ is an $(n-1)\times(n-1)$ reflector such that

$$\tilde{Q}_1 \begin{bmatrix} a_{21} \\ a_{31} \\ \vdots \\ a_{n1} \end{bmatrix} = \begin{bmatrix} * \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

If we multiply $A$ on the left by $Q_1$, we obtain a matrix $Q_1A$ that has zeros in the first column from the third entry on. To complete a similarity transformation, we must multiply on the right by $Q_1$ $(= Q_1^{-1})$. Because $Q_1$ has the form (3), the transformation $Q_1A \mapsto Q_1AQ_1$ does not alter the first column. Thus $Q_1AQ_1$ has the desired zeros in the first column.

The second reflector $Q_2$ has the form

$$Q_2 = \begin{bmatrix} 1 & & \\ & 1 & \\ & & \tilde{Q}_2 \end{bmatrix}$$

and is constructed so that $Q_2Q_1AQ_1Q_2$ has zeros in the second column from the fourth position on. This transformation does not disturb the zeros that were created in the first column in the first step. The details are left for the reader (or see [**17**], for example). The third reflector $Q_3$ creates zeros in the third column, and so on. After $n-2$ reflectors, we will have produced a matrix $H = Q_{n-2}\cdots Q_1AQ_1\cdots Q_{n-2}$ in upper Hessenberg form. Letting $Q = Q_1\cdots Q_{n-2}$, we have $Q^{-1} = Q_{n-2}\cdots Q_1$, and $H = Q^{-1}AQ$. ∎

The proof of Theorem 2 is constructive; that is, it gives a finite algorithm (a direct method) for computing $H$ and $Q$. The operations used are those required for setting up and applying reflectors, namely addition, subtraction, multiplication, division, and the extraction of square roots [**6, 17**]. What we really want is an algorithm to get us all the way to upper triangular form, from which we can read off the eigenvalues. But Hessenberg form is as far as we can get with a finite algorithm using the specified operations. If we had a general procedure for creating even one more zero below the main diagonal, we would be able to split the eigenvalue problem into two smaller eigenvalue problems. If we could do this, we would be able to split each of the smaller

problems into still smaller problems and eventually get to triangular form by a finite algorithm. This would imply the existence of a formula for the zeros of a general $n$th degree polynomial, in violation of Galois theory.

The ability to reduce a matrix to upper Hessenberg form is quite useful. If the $QR$ algorithm (2) is initiated with a matrix $A_0$ that is upper Hessenberg, then (except in some special cases that can be avoided) all iterates $A_j$ are upper Hessenberg. This results in much more economical iterations, as both the $QR$ decomposition and the subsequent $RQ$ multiplication can be done in many fewer operations in the Hessenberg case. Thus an initial reduction to upper Hessenberg form is well worth the extra expense. In fact, the algorithm would not be competitive without it.

For Francis's implicitly-shifted $QR$ algorithm, the Hessenberg form is absolutely essential.

**4. FRANCIS'S ALGORITHM.** Suppose we want to find the eigenvalues of some matrix $A \in \mathbb{R}^{n\times n}$. We continue to focus on the real case; the extension to complex matrices is straightforward. We know from Theorem 2 that we can reduce $A$ to upper Hessenberg form, so let us assume from the outset that $A$ is upper Hessenberg. We can assume, moreover, that $A$ is *properly* upper Hessenberg. This means that all of the subdiagonal entries of $A$ are nonzero: $a_{j+1,j} \neq 0$ for $j = 1, 2, \ldots, n-1$. Indeed, if $A$ is not properly upper Hessenberg, say $a_{k+1,k} = 0$, then $A$ has the block-triangular form

$$A = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix},$$

where $A_{11}$ is $k \times k$. Thus the eigenvalue problem for $A$ reduces to eigenvalue problems for the smaller upper Hessenberg matrices $A_{11}$ and $A_{22}$. If either of these is not properly upper Hessenberg, we can break it apart further until we are finally left with proper upper Hessenberg matrices. We will assume, therefore, that $A$ is properly upper Hessenberg from the outset.

An iteration of Francis's algorithm of degree $m$ begins by picking $m$ *shifts* $\rho_1, \ldots, \rho_m$. In principal $m$ can be any positive integer, but in practice it should be fairly small. Francis took $m = 2$. The rationale for shift selection will be explained later. There are many reasonable ways to choose shifts, but the simplest is to take $\rho_1, \ldots, \rho_m$ to be the eigenvalues of the $m \times m$ submatrix in the lower right-hand corner of $A$. This is an easy computation if $m = 2$.

If $m > 1$, this shifting strategy can produce complex shifts, but they always occur in conjugate pairs. Whether we use this or some other strategy, we will always insist that it have this property: when the matrix is real, complex shifts must be produced in conjugate pairs.

Now let

$$p(A) = (A - \rho_m I) \cdots (A - \rho_1 I).$$

We do not actually compute $p(A)$, as this would be too expensive. Francis's algorithm just needs the first column

$$x = p(A)e_1,$$

which is easily computed by $m$ successive matrix-vector multiplications. The computation is especially cheap because $A$ is upper Hessenberg. It is easy to check that $(A - \rho_1 I)e_1$ has nonzero entries in only its first two positions, $(A - \rho_2 I)(A - \rho_1 I)e_1$ has nonzero entries in only its first three positions, and so on. As a consequence, $x$ has nonzero entries in only its first $m + 1$ positions. The entire computation of $x$ depends

on only the first $m$ columns of $A$ (and the shifts) and requires negligible computational effort if $m$ is small. Since the complex shifts occur in conjugate pairs, $p(A)$ is real. Therefore $x$ is real.

Let $\tilde{x} \in \mathbb{R}^{m+1}$ denote the vector consisting of the first $m+1$ entries of $x$ (the nonzero part). Theorem 1 guarantees that there is an $(m+1) \times (m+1)$ reflector $\tilde{Q}_0$ such that $\tilde{Q}_0 \tilde{x} = \alpha e_1$, where $\alpha = \pm \|\tilde{x}\|_2$. Let $Q_0$ be an $n \times n$ reflector given by

$$Q_0 = \begin{bmatrix} \tilde{Q}_0 & \\ & I \end{bmatrix}. \tag{4}$$

Clearly $Q_0 x = \alpha e_1$ and, since $Q_0$ is an involution, $Q_0 e_1 = \alpha^{-1} x$. Thus the first column of $Q_0$ is proportional to $x$.

Now use $Q_0$ to perform a similarity transformation: $A \mapsto Q_0^{-1} A Q_0 = Q_0 A Q_0$. This disturbs the Hessenberg form but only slightly. Because $Q_0$ has the form (4), the transformation $A \mapsto Q_0 A$ affects only the first $m+1$ rows. Typically the first $m+1$ rows are completely filled in. The transformation $Q_0 A \mapsto Q_0 A Q_0$ affects only the first $m+1$ columns. Since $a_{m+2,m+1} \neq 0$, row $m+2$ gets filled in by this transformation. Below row $m+2$, we have a big block of zeros, and these remain zero. The total effect is that there is a bulge in the Hessenberg form. In the case $m = 3$ and $n = 7$, the matrix $Q_0 A Q_0$ looks like

$$\begin{bmatrix} * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ & & & & * & * & * \\ & & & & & * & * \end{bmatrix}.$$

In this $7 \times 7$ matrix the bulge looks huge. If you envision, say, the $100 \times 100$ case (keeping $m = 3$), you will see that the matrix is almost upper Hessenberg with a tiny bulge at the top.

The rest of the Francis iteration consists of returning this matrix to upper Hessenberg form by the algorithm sketched in the proof of Theorem 2. This begins with a transformation $Q_1$ that acts only on rows 2 through $n$ and creates the desired zeros in the first column. Since the first column already consists of zeros after row $m+2$, the scope of $Q_1$ can be restricted a lot further in this case. It needs to act on rows 2 through $m+2$ and create zeros in positions $(3,1), \ldots, (m+2,1)$. Applying $Q_1$ on the left, we get a matrix $Q_1 Q_0 A Q_0$ that has the Hessenberg form restored in the first column. Completing the similarity transformation, we multiply by $Q_1$ on the right. This recombines columns 2 through $m+2$. Since $a_{m+3,m+2} \neq 0$, additional nonzero entries are created in positions $(m+3,2), \ldots, (m+3, m+1)$. In the case $m = 3$ and $n = 7$ the matrix $Q_1 Q_0 A Q_0 Q_1$ looks like

$$\begin{bmatrix} * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ & * & * & * & * & * & * \\ & * & * & * & * & * & * \\ & * & * & * & * & * & * \\ & * & * & * & * & * & * \\ & & & & & * & * \end{bmatrix}.$$

The bulge has not gotten any smaller, but it has been pushed one position to the right and downward. This establishes the pattern for the process. The next transformation will push the bulge over and down one more position, and so on. Thinking again of the $100 \times 100$ case, we see that a long sequence of such transformations will chase the bulge down through the matrix until it is finally pushed off the bottom. At this point, Hessenberg form will have been restored and the Francis iteration will be complete. For obvious reasons, Francis's algorithm is sometimes referred to as a *bulge chasing* algorithm.

An iteration of Francis's algorithm of degree $m$ can be summarized briefly as follows.

1. Pick some shifts $\rho_1, \ldots, \rho_m$.
2. Compute $x = p(A)e_1 = (A - \rho_m I) \cdots (A - \rho_1 I)e_1$.
3. Compute a reflector $Q_0$ whose first column is proportional to $x$.
4. Do a similarity transformation $A \mapsto Q_0 A Q_0$, creating a bulge.
5. Return the matrix to upper Hessenberg form by chasing the bulge.

Notice that, although this algorithm is commonly known as the implicitly-shifted *QR* algorithm, its execution does not require any *QR* decompositions. Why, then, should we call it the *QR* algorithm? The name is misleading, and for this reason I prefer to call it Francis's algorithm.

Letting $\hat{A}$ denote the final result of the Francis iteration, we have

$$\hat{A} = Q_{n-2} \cdots Q_1 Q_0 A Q_0 Q_1 \cdots Q_{n-2},$$

where $Q_0$ is the transformation that creates the bulge, and $Q_1, \ldots, Q_{n-2}$ are the transformations that chase it. Letting

$$Q = Q_0 Q_1 \cdots Q_{n-2},$$

we have

$$\hat{A} = Q^{-1} A Q.$$

Recall that $Q_0$ was built in such a way that $Q_0 e_1 = \beta x = \beta p(A)e_1$. Looking back to the proof of Theorem 2, we recall that each of the other $Q_i$ has $e_1$ as its first column. Thus $Q_i e_1 = e_1$, $i = 1, \ldots, n-2$. We conclude that

$$Q e_1 = Q_0 Q_1 \cdots Q_{n-2} e_1 = Q_0 e_1 = \beta x.$$

We summarize these findings in a theorem.

**Theorem 3.** *A Francis iteration of degree m with shifts $\rho_1, \ldots, \rho_m$ effects an orthogonal similarity transformation*

$$\hat{A} = Q^{-1} A Q,$$

*where*

$$Q e_1 = \beta p(A) e_1 = \beta (A - \rho_m I) \cdots (A - \rho_1 I) e_1$$

*for some nonzero $\beta$. In words, the first column of $Q$ is proportional to the first column of $p(A)$.*

In the next section we will see why this procedure is so powerful. Repeated Francis iterations with well-chosen shifts typically result in rapid convergence, in the sense that $a_{n-m+1,n-m} \to 0$ quadratically.[4] In a few iterations, $a_{n-m+1,n-m}$ will be small enough to be considered zero. We can then *deflate* the problem:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}.$$

The (small) $m \times m$ matrix $A_{22}$ can be resolved into $m$ eigenvalues with negligible work. (Think of the case $m = 2$, for example.) We can then focus on the remaining $(n-m) \times (n-m)$ submatrix $A_{11}$ and go after another set of $m$ eigenvalues.

## 5. WHY FRANCIS'S ALGORITHM WORKS.

**Subspace Iteration.** At the core of Francis's algorithm lies the humble power method. In the $k$-dimensional version, which is known as *subspace iteration*, we pick a $k$-dimensional subspace $\mathcal{S}$ of $\mathbb{R}^n$ (or perhaps $\mathbb{C}^n$) and, through repeated multiplication by $A$, build the sequence of subspaces

$$\mathcal{S},\ A\mathcal{S},\ A^2\mathcal{S},\ A^3\mathcal{S},\ \ldots. \tag{5}$$

Here by $A\mathcal{S}$ we mean $\{Ax \mid x \in \mathcal{S}\}$. To avoid complications in the discussion, we will make some simplifying assumptions. We will suppose that all of the spaces in the sequence have the same dimension $k$.[5] We also assume that $A$ is a diagonalizable matrix with $n$ linearly independent eigenvectors $v_1, \ldots, v_n$ and associated eigenvalues $\lambda_1, \ldots, \lambda_n$.[6] Sort the eigenvalues and eigenvectors so that $|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n|$. Any vector $x$ can be expressed as a linear combination

$$x = c_1 v_1 + c_2 v_2 + \cdots + c_n v_n$$

for some (unknown) $c_1, \ldots, c_n$. Thus clearly

$$A^j x = c_1 \lambda_1^j v_1 + \cdots + c_k \lambda_k^j v_k + c_{k+1} \lambda_{k+1}^j v_{k+1} + \cdots + c_n \lambda_n^j v_n, \quad j = 1, 2, 3, \ldots.$$

If $|\lambda_k| > |\lambda_{k+1}|$, the components in the directions $v_1, \ldots, v_k$ will grow relative to the components in the directions $v_{k+1}, \ldots, v_n$ as $j$ increases. As a consequence, unless our choice of $\mathcal{S}$ was very unlucky, the sequence (5) will converge to the $k$-dimensional invariant subspace spanned by $v_1, \ldots, v_k$. The convergence is linear with ratio $|\lambda_{k+1}/\lambda_k|$, which means that the error is reduced by a factor of approximately $|\lambda_{k+1}/\lambda_k|$ on each iteration [**18, 15**].

Often the ratio $|\lambda_{k+1}/\lambda_k|$ will be close to 1, so convergence will be slow. In an effort to speed up convergence, we could consider replacing $A$ by some $p(A)$ in (5), giving the iteration

$$\mathcal{S},\ p(A)\mathcal{S},\ p(A)^2\mathcal{S},\ p(A)^3\mathcal{S},\ \ldots. \tag{6}$$

[4]This is a simplification. It often happens that some shifts are much better than others, resulting in $a_{n-k+1,n-k} \to 0$ for some $k < m$.

[5]Of course it can happen that the dimension decreases in the course of the iterations. This does not cause any problems for us. In fact, it is good news [**15**], but we will not discuss that case here.

[6]The non-diagonalizable case is more complicated but leads to similar conclusions. See [**18**] or [**15**], for example.

If $p(z) = (z - \rho_m) \cdots (z - \rho_1)$ is a polynomial of degree $m$, each step of (6) amounts to $m$ steps of (5) with shifts $\rho_1, \ldots, \rho_m$. The eigenvalues of $p(A)$ are $p(\lambda_1), \ldots, p(\lambda_n)$. If we renumber them so that $|p(\lambda_1)| \geq \cdots \geq |p(\lambda_n)|$, the rate of convergence of (6) will be $|p(\lambda_{k+1})/p(\lambda_k)|$. Now we have a bit more flexibility. Perhaps by a wise choice choice of shifts $\rho_1, \ldots, \rho_m$, we can make this ratio small, at least for some values of $k$.

**Subspace Iteration with Changes of Coordinate System.** As we all know, a similarity transformation $\hat{A} = Q^{-1}AQ$ is just a change of coordinate system. $A$ and $\hat{A}$ are two matrices that represent the same linear operator with respect to two different bases. Each vector in $\mathbb{R}^n$ is the coordinate vector of some vector $v$ in the vector space on which the operator acts. If the vector $v$ has coordinate vector $x$ before the change of coordinate system, it will have coordinate vector $Q^{-1}x$ afterwards.

Now consider a step of subspace iteration applied to the special subspace

$$\mathcal{E}_k = \text{span}\{e_1, \ldots, e_k\},$$

where $e_i$ is the standard basis vector with a one in the $i$th position and zeros elsewhere. The vectors $p(A)e_1, \ldots, p(A)e_k$ are a basis for the space $p(A)\mathcal{E}_k$. Let $q_1, \ldots, q_k$ be an orthonormal basis for $p(A)\mathcal{E}_k$, which could be obtained by some variant of the Gram-Schmidt process, for example. Let $q_{k+1}, \ldots, q_n$ be additional orthonormal vectors such that $q_1, \ldots, q_n$ together form an orthonormal basis of $\mathbb{R}^n$, and let $Q = \begin{bmatrix} q_1 & \cdots & q_n \end{bmatrix} \in \mathbb{R}^{n \times n}$. Since $q_1, \ldots, q_n$ are orthonormal, $Q$ is an orthogonal matrix.

Now use $Q$ to make a change of coordinate system

$$\hat{A} = Q^{-1}AQ = Q^TAQ.$$

Let us see what this change of basis does to the space $p(A)\mathcal{E}_k$. To this end we check the basis vectors $q_1, \ldots, q_k$. Under the change of coordinate system, these get mapped to $Q^Tq_1, \ldots, Q^Tq_k$. Because the columns of $Q$ are the orthonormal vectors $q_1, \ldots, q_n$, the vectors $Q^Tq_1, \ldots, Q^Tq_k$ are exactly $e_1, \ldots, e_k$. Thus the change of coordinate system maps $p(A)\mathcal{E}_k$ back to $\mathcal{E}_k$.

Now, if we want to do another step of subspace iteration, we can work in the new coordinate system, applying $p(\hat{A})$ to $\mathcal{E}_k$. Then we can do another change of coordinate system and map $p(\hat{A})\mathcal{E}_k$ back to $\mathcal{E}_k$. If we continue to iterate in this manner, we produce a sequence of orthogonally similar matrices $(A_j)$ through successive changes of coordinate system, and we are always dealing with the same subspace $\mathcal{E}_k$. This is a version of subspace iteration for which the subspace stays fixed and the matrix changes.

What does convergence mean in this case? As $j$ increases, $\mathcal{E}_k$ comes closer and closer to being invariant under $A_j$. The special space $\mathcal{E}_k$ is invariant under $A_j$ if and only if $A_j$ has the block-triangular form

$$A_j = \begin{bmatrix} A_{11}^{(j)} & A_{12}^{(j)} \\ 0 & A_{22}^{(j)} \end{bmatrix},$$

where $A_{11}^{(j)}$ is $k \times k$. Of course, we just approach invariance; we never attain it exactly. In practice we will have

$$A_j = \begin{bmatrix} A_{11}^{(j)} & A_{12}^{(j)} \\ A_{21}^{(j)} & A_{22}^{(j)} \end{bmatrix},$$

where $A_{21}^{(j)} \to 0$ linearly with ratio $|p(\lambda_{k+1})/p(\lambda_k)|$. Eventually $A_{21}^{(j)}$ will get small enough that we can set it to zero and split the eigenvalue problem into two smaller problems.

If one thinks about implementing subspace iteration with changes of coordinate system in a straightforward manner, as outlined above, it appears to be a fairly expensive procedure. Fortunately there is a way to implement this method on upper Hessenberg matrices at reasonable computational cost, namely Francis's algorithm. This amazing procedure effects subspace iteration with changes of coordinate system, not just for one $k$, but for $k = 1, \ldots, n-1$ all at once. At this point we are not yet ready to demonstrate this fact, but we can get a glimpse at what is going on.

Francis's algorithm begins by computing the vector $p(A)e_1$. This is a step of the power method, or subspace iteration with $k = 1$, mapping $\mathcal{E}_1 = \text{span}\{e_1\}$ to $p(A)\mathcal{E}_1$. Then, according to Theorem 3, this is followed by a change of coordinate system $\hat{A} = Q^{-1}AQ$ in which $\text{span}\{q_1\} = p(A)\mathcal{E}_1$. This is exactly the case $k = 1$ of subspace iteration with a change of coordinate system. Thus if we perform iterations repeatedly with the same shifts (the effect of changing shifts will be discussed later), the sequence of iterates $(A_j)$ so produced will converge linearly to the form

$$\left[\begin{array}{c|ccc} \lambda_1 & * & \cdots & * \\ \hline 0 & * & \cdots & * \\ \vdots & \vdots & & \vdots \\ 0 & * & \cdots & * \end{array}\right].$$

Since the iterates are upper Hessenberg, this just means that $a_{21}^{(j)} \to 0$. The convergence is linear with ratio $|p(\lambda_2)/p(\lambda_1)|$.

This is just the tip of the iceberg. To get the complete picture, we must bring one more concept into play.

**Krylov Subspaces.** Given a nonzero vector $x$, the sequence of *Krylov subspaces* associated with $x$ is defined by

$$\begin{aligned} \mathcal{K}_1(A, x) &= \text{span}\{x\}, \\ \mathcal{K}_2(A, x) &= \text{span}\{x, Ax\}, \\ \mathcal{K}_3(A, x) &= \text{span}\{x, Ax, A^2x\}, \end{aligned}$$

and in general $\mathcal{K}_k(A, x) = \text{span}\{x, Ax, \ldots, A^{k-1}x\}$, $k = 1, 2, \ldots, n$. We need to make a couple of observations about Krylov subspaces. The first is that wherever there are Hessenberg matrices, there are Krylov subspaces lurking in the background.

**Theorem 4.** *Let $A$ be properly upper Hessenberg. Then*

$$\text{span}\{e_1, \ldots, e_k\} = \mathcal{K}_k(A, e_1), \quad k = 1, \ldots, n.$$

The proof is an easy exercise.

**Theorem 5.** *Suppose $H = Q^{-1}AQ$, where $H$ is properly upper Hessenberg. Let $q_1, \ldots, q_n$ denote the columns of $Q$. Then*

$$\text{span}\{q_1, \ldots, q_k\} = \mathcal{K}_k(A, q_1), \quad k = 1, \ldots, n.$$

In our application, $Q$ is orthogonal, but this theorem is valid for any nonsingular $Q$. In the special case $Q = I$, it reduces to Theorem 4.

*Proof.* We use induction on $k$. For $k = 1$, the result holds trivially. Now, for the induction step, rewrite the similarity transformation as $AQ = QH$. Equating $k$th columns of this equation we have, for $k = 1, \ldots, n-1$,

$$Aq_k = \sum_{i=1}^{k+1} q_i h_{ik}.$$

The sum stops at $k+1$ because $H$ is upper Hessenberg. We can rewrite this as

$$q_{k+1}h_{k+1,k} = Aq_k - \sum_{i=1}^{k} q_i h_{ik}. \tag{7}$$

Our induction hypothesis is that $\text{span}\{q_1, \ldots, q_k\} = \mathcal{K}_k(A, q_1)$. Since $q_k \in \mathcal{K}_k(A, q_1)$, it is clear that $Aq_k \in \mathcal{K}_{k+1}(A, q_1)$. Then, since $h_{k+1,k} \neq 0$, equation (7) implies that $q_{k+1} \in \mathcal{K}_{k+1}(A, q_1)$. Thus $\text{span}\{q_1, \ldots, q_{k+1}\} \subseteq \mathcal{K}_{k+1}(A, q_1)$. Since $q_1, \ldots, q_{k+1}$ are linearly independent, and the dimension of $\mathcal{K}_{k+1}(A, q_1)$ is at most $k+1$, these subspaces must be equal. ∎

We remark in passing that equation (7) has computational as well as theoretical importance. It is the central equation of the *Arnoldi process*, one of the most important algorithms for large, sparse matrix computations [**1**].

Our second observation about Krylov subspaces is in connection with subspace iteration. Given a matrix $A$, which we take for granted as our given object of study, we can say that each nonzero vector $x$ contains the information to build a whole sequence of Krylov subspaces $\mathcal{K}_1(A, x)$, $\mathcal{K}_2(A, x)$, $\mathcal{K}_3(A, x)$, .... Now consider a step of subspace iteration applied to a Krylov subspace. One easily checks that

$$p(A)\mathcal{K}_k(A, x) = \mathcal{K}_k(A, p(A)x),$$

as a consequence of the equation $Ap(A) = p(A)A$. Thus the result of a step of subspace iteration on a Krylov subspace generated by $x$ is another Krylov subspace, namely the one generated by $p(A)x$. It follows that

> the power method $x \mapsto p(A)x$ contains all the information about a whole sequence of nested subspace iterations,

in the following sense. The vector $x$ generates a whole sequence of Krylov subspaces $\mathcal{K}_k(A, x)$, $k = 1, \ldots, n$, and $p(A)x$ generates the sequence $\mathcal{K}_k(A, p(A)x)$, $k = 1, \ldots, n$. Moreover, the $k$th space $\mathcal{K}_k(A, p(A)x)$ is the result of one step of subspace iteration on the $k$th space $\mathcal{K}_k(A, x)$.

**Back to Francis's Algorithm.** Each iteration of Francis's algorithm executes a step of the power method $e_1 \mapsto p(A)e_1$ followed by a change of coordinate system. As we have just seen, the step of the power method induces subspace iterations on the corresponding Krylov subspaces: $\mathcal{K}_k(A, e_1) \mapsto \mathcal{K}_k(A, p(A)e_1)$, $k = 1, \ldots, n$. This is not just some theoretical action. Since Francis's algorithm operates on properly upper Hessenberg matrices, the Krylov subspaces in question reside in the columns of the transforming matrices.

Indeed, a Francis iteration makes a change of coordinate system $\hat{A} = Q^{-1}AQ$. It can happen that $\hat{A}$ has one or more zeros on the subdiagonal. This is a rare and lucky event, which allows us to reduce the problem immediately to two smaller eigenvalue problems. Let us assume we have not been so lucky. Then $\hat{A}$ is properly upper Hessenberg, and we can apply Theorem 5, along with Theorems 3 and 4, to deduce that, for $k = 1, \ldots, n$,

$$\begin{aligned}\text{span}\{q_1, \ldots, q_k\} &= \mathcal{K}_k(A, q_1) \\ &= \mathcal{K}_k(A, p(A)e_1) \\ &= p(A)\mathcal{K}_k(A, e_1) \\ &= p(A)\text{span}\{e_1, \ldots, e_k\}.\end{aligned}$$

These equations show that Francis's algorithm effects subspace iteration with a change of coordinate system for dimensions $k = 1, \ldots, n-1$.

Now suppose we pick some shifts $\rho_1, \ldots, \rho_m$, let $p(z) = (z - \rho_m) \cdots (z - \rho_1)$, and do a sequence of Francis iterations with this fixed $p$ to produce the sequence $(A_j)$. Then, for each $k$ for which $|p(\lambda_{k+1})/p(\lambda_k)| < 1$, we have $a_{k+1,k}^{(j)} \to 0$ linearly with rate $|p(\lambda_{k+1})/p(\lambda_k)|$. Thus all of the ratios

$$|p(\lambda_{k+1})/p(\lambda_k)|, \quad k = 1, \ldots, n-1,$$

are important. If any one of them is small, then one of the subdiagonal entries will converge rapidly to zero, allowing us to break the problem into smaller eigenvalue problems.

**Choice of Shifts.** The convergence results we have so far are based on the assumption that we are going to pick some shifts $\rho_1, \ldots, \rho_m$ in advance and use them over and over again. Of course, this is not what really happens. In practice, we get access to better and better shifts as we proceed, so we might as well use them. How, then, does one choose good shifts?

To answer this question we start with a thought experiment. Suppose that we are somehow able to find $m$ shifts that are excellent in the sense that each of them approximates one eigenvalue well (good to several decimal places, say) and is not nearly so close to any of the other eigenvalues. Assume the $m$ shifts approximate $m$ different eigenvalues. We have

$$p(z) = (z - \rho_m) \cdots (z - \rho_1),$$

where each zero of $p$ approximates an eigenvalue well. Then, for each eigenvalue $\lambda_k$ that is well approximated by a shift, $|p(\lambda_k)|$ will be tiny. There are $m$ such eigenvalues. For each eigenvalue that is not well approximated by a shift, $|p(\lambda_k)|$ will not be small. Thus, exactly $m$ of the numbers $|p(\lambda_k)|$ are small. If we renumber the eigenvalues so that $|p(\lambda_1)| \geq |p(\lambda_2)| \geq \cdots \geq |p(\lambda_n)|$, we will have $|p(\lambda_{n-m})| \gg |p(\lambda_{n-m+1})|$, and

$$|p(\lambda_{n-m+1})/p(\lambda_{n-m})| \ll 1.$$

This will cause $a_{n-m+1,n-m}^{(j)}$ to converge to zero rapidly. Once this entry gets very small,

the bottom $m \times m$ submatrix

$$\begin{bmatrix} a^{(j)}_{n-m+1,n-m+1} & \cdots & a^{(j)}_{n-m+1,n} \\ \vdots & & \vdots \\ a^{(j)}_{n,n-m+1} & \cdots & a^{(j)}_{n,n} \end{bmatrix} \tag{8}$$

will be nearly separated from the rest of the matrix. Its eigenvalues will be excellent approximations to eigenvalues of $A$, eventually even substantially better than the shifts that were used to find them.[7] It makes sense, then, to take the eigenvalues of (8) as new shifts, replacing the old ones. This will result in a reduction of the ratio $|p(\lambda_{n-m+1})/p(\lambda_{n-m})|$ and even faster convergence. At some future point it would make sense to replace these shifts by even better ones to get even better convergence.

This thought experiment shows that at some point it makes sense to use the eigenvalues of (8) as shifts, but several questions remain. How should the shifts be chosen initially? At what point should we switch to the strategy of using eigenvalues of (8) as shifts? How often should we update the shifts?

The answers to these questions were not at all evident to Francis and the other eigenvalue computation pioneers. A great deal of testing and experimentation has led to the following empirical conclusions. The shifts should be updated on every iteration, and it is okay to use the eigenvalues of (8) as shifts right from the very start. At first they will be poor approximations, and progress will be slow. After a few (or sometimes more) iterations, they will begin to home in on eigenvalues and the convergence rate will improve. With new shifts chosen on each iteration, the method normally converges quadratically [**18, 15**], which means that $|a^{(j+1)}_{n-m+1,n-m}|$ will be roughly the square of $|a^{(j)}_{n-m+1,n-m}|$. Thus successive values of $|a^{(j)}_{n-m+1,n-m}|$ could be approximately $10^{-3}$, $10^{-6}$, $10^{-12}$, $10^{-24}$, for example. Very quickly this number gets small enough that we can declare it to be zero and split off an $m \times m$ submatrix and $m$ eigenvalues. Then we can deflate the problem and go after the next set of $m$ eigenvalues.

There is one caveat. The strategy of using the eigenvalues of (8) as shifts does not always work, so variants have been introduced. And it is safe to say that all shifting strategies now in use are indeed variants of this simple strategy. The stategies used by the codes in MATLAB and other modern software are quite good, but they might not be unbreakable. Certainly nobody has been able to prove that they are. It is an open question to come up with a shifting strategy that provably always works and normally yields rapid convergence.

It is a common phenomenon that numerical methods work better than we can prove they work. In Francis's algorithm the introduction of dynamic shifting (new shifts on each iteration) dramatically improves the convergence rate but at the same time makes the analysis much more difficult.

**Next-to-Last Words.** Francis's algorithm is commonly known as the implicitly-shifted *QR* algorithm, but it bears no resemblance to the basic *QR* algorithm (2). We have described Francis's algorithm and explained why it works without referring to (2) in any way.

One might eventually like to see the connection between Francis's algorithm and the *QR* decomposition:

[7] The eigenvalues that are well approximated by (8) are indeed the same as the eigenvalues that are well approximated by the shifts.

**Theorem 6.** *Consider a Francis iteration*

$$\hat{A} = Q^{-1}AQ$$

*of degree m with shifts* $\rho_1, \ldots, \rho_m$. *Let* $p(z) = (z - \rho_m) \cdots (z - \rho_1)$, *as usual. Then there is an upper-triangular matrix R such that*

$$p(A) = QR.$$

*In words, the transforming matrix Q of a Francis iteration is exactly the orthogonal factor in the QR decomposition of* $p(A)$.

The proof can be found in [**17**], for example. Theorem 6 is also valid for a long sequence of Francis iterations, for which $\rho_1, \ldots, \rho_m$ is the long list of all shifts used in all iterations ($m$ could be a million). This is a useful tool for formal convergence proofs. In fact, all formal convergence results for *QR* and related algorithms that I am aware of make use of this tool or a variant of it.

In this paper we have focused on computation of eigenvalues. However, Francis's algorithm can be adapted in straightforward ways to the computation of eigenvectors and invariant subspaces as well [**6, 15, 17**].

**6. WHAT BECAME OF JOHN FRANCIS?** From about 1970 on, Francis's algorithm was firmly established as *the* most important algorithm for the eigenvalue problem. Francis's two papers [**3, 4**] accumulated hundreds of citations. As a part of the celebration of the arrival of the year 2000, Dongarra and Sullivan [**2**] published a list of the top ten algorithms of the twentieth century. "The *QR* algorithm" was on that list. Francis's work had become famous, but by the year 2000 nobody in the community of numerical analysts knew what had become of the man or could recall ever having met him. It was even speculated that he had died.

Two members of our community, Gene Golub and Frank Uhlig, working independently, began to search for him. With the help of the internet they were able to find him, alive and well, retired in the South of England. In August of 2007 Golub visited Francis at his home. This was just a few months before Gene's untimely death in November. Golub reported that Francis had been completely unaware of the huge impact of his work. The following summer Uhlig was able to visit Francis and chat with him several times over the course of a few days. See Uhlig's papers [**5, 14**] for more information about Francis's life and career.

Uhlig managed to persuade Francis to attend and speak at the Biennial Numerical Analysis conference at the University of Strathclyde, Glasgow, in June of 2009. With some help from Andy Wathen, Uhlig organized a minisymposium at that conference honoring Francis and discussing his work and the related foundational work of Rutishauser and Kublanovskaya. I had known of the Biennial Numerical Analysis meetings in Scotland for many years and had many times thought of attending, but I had never actually gotten around to doing so. John Francis changed all that. No way was I going to pass up the opportunity to meet the man and speak in the same minisymposium with him. As the accompanying photograph shows, Francis remains in great shape at age 75. His mind is sharp, and he was able to recall a lot about the *QR* algorithm and how it came to be. We enjoyed his reminiscences very much.

A few months later I spoke about Francis and his algorithm at the Pacific Northwest Numerical Analysis Seminar at the University of British Columbia. I displayed this

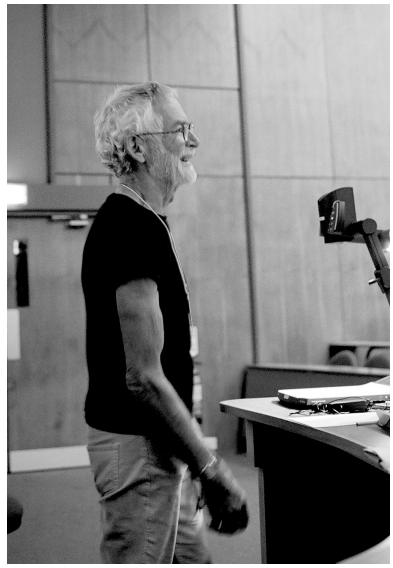

**Figure 1.** John Francis speaks at the University of Strathclyde, Glasgow, in June 2009. Photo: Frank Uhlig.

photograph and remarked that Francis has done a better job than I have at keeping his weight under control over the years. Afterward Randy LeVeque explained it to me: Francis knows how to chase the bulge.

REFERENCES


1. Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, eds., *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, Society for Industrial and Applied Mathematics, Philadelphia, 2000.
2. J. Dongarra and F. Sullivan, The top 10 algorithms, *Comput. Sci. Eng.* **2** (2000) 22–23. doi:10.1109/MCISE.2000.814652
3. J. G. F. Francis, The *QR* transformation, part I, *Computer J.* **4** (1961) 265–272. doi:10.1093/comjnl/4.3.265
4. ———, The *QR* transformation, part II, *Computer J.* **4** (1961) 332–345. doi:10.1093/comjnl/4.4.332
5. G. H. Golub and F. Uhlig, The *QR* algorithm: 50 years later its genesis by John Francis and Vera Kublanovskaya and subsequent developments, *IMA J. Numer. Anal.* **29** (2009) 467–485. doi:10.1093/imanum/drp012
6. G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed., Johns Hopkins University Press, Baltimore, MD, 1996.
7. R. Granat, B. Kågström, and D. Kressner, A novel parallel *QR* algorithm for hybrid distributed memory HPC systems, *SIAM J. Sci. Stat. Comput.* **32** (2010) 2345–2378. doi:10.1137/090756934
8. M. Gutknecht and B. Parlett, From qd to *LR*, or, How were the qd and *LR* algorithms discovered? *IMA J. Numer. Anal.* (to appear).
9. J. Hadamard, Essai sur l'étude des fonctions données par leur développement de Taylor, *J. Math.* **4** (1892) 101–182.
10. V. N. Kublanovskaya, On some algorithms for the solution of the complete eigenvalue problem, *USSR Comput. Math. and Math. Phys.* **1** (1962) 637–657. doi:10.1016/0041-5553(63)90168-X
11. H. Rutishauser, Der Quotienten-Differenzen-Algorithmus, *Z. Angew. Math. Phys.* **5** (1954) 233–251. doi:10.1007/BF01600331
12. ———, *Der Quotienten-Differenzen-Algorithmus*, Mitt. Inst. Angew. Math. ETH, no. 7, Birkhäuser, Basel, 1957.
13. ———, Solution of eigenvalue problems with the *LR*-transformation, *Nat. Bur. Standards Appl. Math. Ser.* **49** (1958) 47–81.

14. F. Uhlig, Finding John Francis who found *QR* fifty years ago, *IMAGE, Bulletin of the International Linear Algebra Society* **43** (2009) 19–21; available at http://www.ilasic.math.uregina.ca/iic/IMAGE/IMAGES/image43.pdf.
15. D. S. Watkins, *The Matrix Eigenvalue Problem: G R and Krylov Subspace Methods*, Society for Industrial and Applied Mathematics, Philadelphia, 2007.
16. ———, The *QR* algorithm revisited, *SIAM Rev.* **50** (2008) 133–145. doi:10.1137/060659454
17. ———, *Fundamentals of Matrix Computations*, 3rd ed., John Wiley, Hoboken, NJ, 2010.
18. D. S. Watkins and L. Elsner, Convergence of algorithms of decomposition type for the eigenvalue problem, *Linear Algebra Appl.* **143** (1991) 19–47. doi:10.1016/0024-3795(91)90004-G
19. J. H. Wilkinson, The evaluation of the zeros of ill-conditioned polynomials, parts I and II, *Numer. Math.* **1** (1959) 150–166, 167–180. doi:10.1007/BF01386381
20. ———, *The Algebraic Eigenvalue Problem*, Oxford University Press, Oxford, 1965.
21. ———, The perfidious polynomial, in *Studies in Numerical Analysis*, MAA Studies in Mathematics, vol. 24, G. H. Golub, ed., Mathematical Association of America, Washington, DC, 1984, 1–28.

**DAVID S. WATKINS** received his B.A. from the University of California at Santa Barbara in 1970, his M.Sc. from the University of Toronto in 1971, and his Ph.D. from the University of Calgary in 1974 under the supervision of Peter Lancaster. He enjoys figuring out mathematics and explaining it to others in the simplest and clearest terms. It happens that the mathematics that most frequently draws his attention is related to matrix computations.

*Department of Mathematics, Washington State University, Pullman, WA 99164-3113*
*watkins@wsu.edu*

## The Role of Definitions in Mathematics

On p. 568 of the June–July 2010 issue of this MONTHLY, Oliver Heaviside is quoted as saying "Mathematics is an experimental science, and definitions do not come first, but later on." It is interesting to compare this to Alfred Tarski's view of definitions:

> In fact, I am rather inclined to agree with those who maintain that the moments of greatest creative advancement in science frequently coincide with the introduction of new notions by means of definition.
>
> Alfred Tarski, The semantic conception of truth and the foundations of semantics, *Philosophy and Phenomenological Research* **4** (1944), p. 359

—Submitted by John L. Leonard, University of Arizona