1 Eigenvalues and eigenvectors

1.1 Introduction

A non-zero column-vector $\mathbf{v}$ is called the *eigenvector* of a matrix A with the *eigenvalue* λ, if

$$\mathrm{A}\mathbf{v} = \lambda\mathbf{v}\,. \tag{1}$$

If an $n \times n$ matrix A is real and symmetric, $\mathrm{A}^\mathsf{T} = \mathrm{A}$, then it has n real eigenvalues $\lambda_1, \ldots, \lambda_n$, and its (orthogonalized) eigenvectors $\mathrm{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ form a full basis,

$$\mathrm{V}\mathrm{V}^\mathsf{T} = \mathrm{V}^\mathsf{T}\mathrm{V} = \mathbf{1}\,, \tag{2}$$

in which the matrix is diagonal,

$$\mathrm{V}^\mathsf{T}\mathrm{A}\mathrm{V} = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & & \vdots \\ \vdots & & \ddots & \\ 0 & \cdots & & \lambda_n \end{bmatrix} \equiv \mathrm{D}\,. \tag{3}$$

Matrix diagonalization means finding all eigenvalues and (optionally) eigenvectors of a matrix. Once all eigenvalues and eigenvectors are found, the *Eigenvalue Decomposition* (EVD) of the matrix is given as

$$\mathrm{A} = \mathrm{V}\mathrm{D}\mathrm{V}^\mathsf{T}\,. \tag{4}$$

Eigenvalues and eigenvectors enjoy a multitude of applications in different branches of science and technology.

1.2 Similarity transformations

Orthogonal transformations,

$$\mathrm{A} \to \mathrm{Q}^\mathsf{T}\mathrm{A}\mathrm{Q}\,, \tag{5}$$

where $Q^T Q = \mathbf{1}$, and, generally, similarity transformations,

$$\mathrm{A} \to \mathrm{S}^{-1}\mathrm{A}\mathrm{S}\,, \tag{6}$$

preserve eigenvalues and eigenvectors. Therefore one of the strategies to diagonalize a matrix is to apply a sequence of similarity transformations (also called rotations) which (iteratively) turn the matrix into diagonal form.

1.2.1 Jacobi eigenvalue algorithm

Jacobi eigenvalue algorithm is an iterative method to calculate the eigenvalues and eigenvectors of a real symmetric matrix by a sequence of Jacobi rotations.

Jacobi rotation is an orthogonal transformation which zeroes a pair of the off-diagonal elements of a (real symmetric) matrix A,

$$\mathrm{A} \rightarrow \mathrm{A}' = \mathrm{J}(p,q)^{\mathsf{T}} \mathrm{A} \mathrm{J}(p,q) \ : \ A'_{pq} = A'_{qp} = 0 \ . \tag{7}$$

The orthogonal matrix $\mathrm{J}(p,q)$ which eliminates the element A_{pq} is called the Jacobi rotation matrix. It is equal identity matrix except for the four elements with indices pp, pq, qp, and qq,

$$\mathrm{J}(p,q) = \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & 0 & \\ & & \cos\theta & \cdots & \sin\theta & & \\ & & \vdots & \ddots & \vdots & & \\ & & -\sin\theta & \cdots & \cos\theta & & \\ & 0 & & & & \ddots & \\ & & & & & & 1 \end{bmatrix} \begin{matrix} \\ \\ \leftarrow \text{row } p \\ \\ \leftarrow \text{row } q \\ \\ \\ \end{matrix} . \tag{8}$$

Or explicitly,

$$\begin{aligned} & J(p,q)_{ij} = \delta_{ij} \ \forall \ ij \notin \{pq, qp, pp, qq\} \ ; \\ & J(p,q)_{pp} = \cos\theta = J(p,q)_{qq} \ ; \\ & J(p,q)_{pq} = \sin\theta = -J(p,q)_{qp} \ . \end{aligned} \tag{9}$$

After a Jacobi rotation, $\mathrm{A} \rightarrow \mathrm{A}' = \mathrm{J}^{\mathsf{T}} \mathrm{A} \mathrm{J}$, the matrix elements of A' become

$$\begin{aligned} A'_{ij} &= A_{ij} \ \forall \ i \neq p,q \wedge j \neq p,q \\ A'_{pi} &= A'_{ip} = cA_{pi} - sA_{qi} \ \forall \ i \neq p,q \ ; \\ A'_{qi} &= A'_{iq} = sA_{pi} + cA_{qi} \ \forall \ i \neq p,q \ ; \\ A'_{pp} &= c^2 A_{pp} - 2scA_{pq} + s^2 A_{qq} \ ; \\ A'_{qq} &= s^2 A_{pp} + 2scA_{pq} + c^2 A_{qq} \ ; \\ A'_{pq} &= A'_{qp} = sc(A_{pp} - A_{qq}) + (c^2 - s^2) A_{pq} \ , \end{aligned} \tag{10}$$

where $c \equiv \cos\theta$, $s \equiv \sin\theta$. The angle θ is chosen such that after rotation the matrix element A'_{pq} is zeroed,

$$\tan(2\theta) = \frac{2A_{pq}}{A_{qq} - A_{pp}} \ \Rightarrow \ A'_{pq} = 0 \ , \theta = \frac{1}{2} \mathrm{atan2}(2A_{pq}, A_{qq} - A_{pp}) \ , \tag{11}$$

where the atan2 correctly deals with the cases where one or two arguments are equal zero.

A side effect of zeroing a given off-diagonal element A_{pq} by a Jacobi rotation is that other off-diagonal elements are changed. Namely, the elements of the rows and columns with indices p and q. However, after the Jacobi rotation the sum of squares of all off-diagonal elements is reduced. The algorithm repeatedly performs rotations until the off-diagonal elements become sufficiently small.

The convergence of the Jacobi method can be proved for two strategies for choosing the order in which the elements are zeroed:

1. *Classical method*: with each rotation the largest of the remaining off-diagonal elements is zeroed.

2. *Cyclic method*: the off-diagonal elements are zeroed in strict order, e.g. row after row.

Although the classical method allows the least number of rotations, it is typically slower than the cyclic method since searching for the largest element is an $O(n^2)$ operation. The count can be reduced by keeping an additional array with indexes of the largest elements in each row. Updating this array after each rotation is only an $O(n)$ operation.

A *sweep* is a sequence of Jacobi rotations applied to all non-diagonal elements. Typically the method converges after a small number of sweeps. The operation count is $O(n)$ for a Jacobi rotation and $O(n^3)$ for a sweep.

The typical convergence criterion is that the diagonal elements have not changed after a sweep. Other criteria can also be used, like the sum of absolute values of the off-diagonal elements is small, $\sum_{i<j} |A_{ij}| < \epsilon$, where ϵ is the required accuracy, or the largest off-diagonal element is small, $\max |A_{i<j}| < \epsilon$.

Eigenvectors Once the matrix A is diagonalized,

$$\mathrm{Q}^{\mathsf{T}}\mathrm{AQ} = \mathrm{D} \ , \tag{12}$$

the matrix Q becomes the matrix of eigenvectors. The latter can therefore be calculated as $V = \mathbf{1}\mathrm{J}_0\mathrm{J}_2...$, where J_i are the successive Jacobi matrices. At each iteration the update of the V-matrix is given as

$$\begin{aligned} V_{ij} &\rightarrow V_{ij} \ , \ j \neq p, q \\ V_{ip} &\rightarrow cV_{ip} - sV_{iq} \\ V_{iq} &\rightarrow sV_{ip} + cV_{iq} \end{aligned} \tag{13}$$

Alternatively, if only one (or few) eigenvector $\mathbf{v}_k$ is needed, one can instead solve the (singular) system $(A - \lambda_k)\mathbf{v} = 0$.

Ordering of eigenvalues Suppose the matrix element A_{pq} to be zeroed by a Jacobi rotation is already zero. Then the function

$$\theta = \frac{1}{2}\mathrm{atan2}(2A_{pq}, A_{qq} - A_{pp}) \tag{14}$$

will return 0 if $A_{qq} > A_{pp}$ or $\pi/2$ if $A_{qq} < A_{pp}$. In the first case no rotation will take place, while in the second case the rotation with $\theta = \pi/2$ will be applied. The latter will exchange the matrix elements A_{qq} and A_{pp}. That is, the diagonal elements will be arranged in accending order.

1.2.2 QR/QL algorithm

An orthogonal transformation of a real symmetric matrix, $\mathrm{A} \to \mathrm{Q}^T\mathrm{AQ} = \mathrm{RQ}$, where Q is from the QR-decomposition of A, partly turns the matrix A into diagonal form. Successive iterations eventually make it diagonal. If there are degenerate eigenvalues there will be a corresponding block-diagonal sub-matrix.

For convergence properties it is of advantage to use *shifts*: instead of QR[A] we do $\mathrm{QR}[\mathrm{A} - s\mathbf{1}]$ and then $\mathrm{A} \to \mathrm{RQ} + s\mathbf{1}$. The shift s can be chosen as A_{nn}. As soon as an eigenvalue is found the matrix is deflated, that is, the corresponding row and column are crossed out.

Accumulating the successive transformation matrices Q_i into the total matrix $\mathrm{Q} = \mathrm{Q}_1 \ldots \mathrm{Q}_N$, such that $Q^T\mathrm{AQ} = \Lambda$, gives the eigenvectors as columns of the Q matrix.

If only one (or few) eigenvector $\mathbf{v}_k$ is needed one can instead solve the (singular) system $(\mathrm{A} - \lambda_k)\mathbf{v} = 0$.

Tridiagonalization. Each iteration of the QR/QL algorithm is an $O(n^3)$ operation. On a tridiagonal matrix it is only $O(n)$. Therefore the effective strategy is first to make the matrix tridiagonal and then apply the QR/QL algorithm. Tridiagonalization of a matrix is a non-iterative operation with a fixed number of steps.

1.3 Eigenvalues of updated matrix

In practice it happens quite often that the matrix A to be diagonalized is given in the form of a diagonal matrix, D, plus an update matrix, W,

$$\mathrm{A} = \mathrm{D} + \mathrm{W}\,, \tag{15}$$

where the update W is a simpler, in a certain sense, matrix which allows a more efficient calculation of the updated eigenvalues, as compared to general diagonalization algorithms.

The most common updates are

- symmetric rank-1 update,
$$\mathrm{W} = \mathbf{u}\mathbf{u}^{\mathsf{T}}\,, \tag{16}$$
where $\mathbf{u}$ is a columnt-vector;
- symmetric rank-2 update,
$$\mathrm{W} = \mathbf{u}\mathbf{v}^{\mathsf{T}} + \mathbf{v}\mathbf{u}^{\mathsf{T}}\,; \tag{17}$$

- symmetric row/column update – a special case of rank-2 update,

$$\mathrm{W} = \begin{bmatrix} 0 & \dots & u_1 & \dots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ u_1 & \dots & u_p & \dots & u_n \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \dots & u_n & \dots & 0 \end{bmatrix} \equiv \mathbf{e}(p)\mathbf{u}^{\mathsf{T}} + \mathbf{u}\mathbf{e}(p)^{\mathsf{T}} \,, \tag{18}$$

  where $\mathbf{e}(p)$ is the unit vector in the p-direction.

1.3.1 Rank-1 update

We assume that a size-n real symmetric matrix A to be diagonalized is given in the form of a diagonal matrix plus a rank-1 update,

$$\mathrm{A} = \mathrm{D} + \sigma\mathbf{u}\mathbf{u}^{\mathsf{T}} \,, \tag{19}$$

where D is a diagonal matrix with diagonal elements $\{d_1, \dots, d_n\}$ and $\mathbf{u}$ is a given vector. The diagonalization of such matrix can be done in $O(m^2)$ operations, where $m \leq n$ is the number of non-zero elements in the update vector $\mathbf{u}$, as compared to $O(n^3)$ operations for a general diagonalization [2].

The eigenvalue equation for the updated matrix reads

$$\left(\mathrm{D} + \sigma\mathbf{u}\mathbf{u}^{\mathsf{T}}\right)\mathbf{q} = \lambda\mathbf{q} \,, \tag{20}$$

where λ is an eigenvalue and $\mathbf{q}$ is the corresponding eigenvector. The equation can be rewritten as

$$\left(\mathrm{D} - \lambda 1\right)\mathbf{q} + \sigma\mathbf{u}\mathbf{u}^{\mathsf{T}}\mathbf{q} = 0 \,. \tag{21}$$

Multiplying from the left with $\mathbf{u}^{\mathsf{T}}\left(\mathrm{D} - \lambda 1\right)^{-1}$ gives

$$\mathbf{u}^{\mathsf{T}}\mathbf{q} + \mathbf{u}^{\mathsf{T}}\left(\mathrm{D} - \lambda 1\right)^{-1}\sigma\mathbf{u}\mathbf{u}^{\mathsf{T}}\mathbf{q} = 0 \,. \tag{22}$$

Finally, dividing by $\mathbf{u}^{\mathsf{T}}\mathbf{q}$ leads to the (scalar) *secular equation* (or *characteristic equation*) in λ,

$$1 + \sum_{i=1}^{m} \frac{\sigma u_i^2}{d_i - \lambda} = 0 \,, \tag{23}$$

where the summation index counts the m non-zero components of the update vector $\mathbf{u}$. The m roots of this equation determine the (updated) eigenvalues[1].

Finding a root of a rational function requires an iterative technique, such as the Newton-Raphson method. Therefore diagonalization of an updated matrix is still an iterative procedure. However, each root can be found in $O(1)$ iterations, each

[1] Multiplying this equation by $\prod_{i=1}^{m}(d_i - \lambda)$ leads to an equivalent polynomial equation of the order m, which has exactly m roots.

iteration requiring $O(m)$ operations. Therefore the iterative part of this algorithm — finding all m roots — needs $O(m^2)$ operations.

Finding roots of this particular secular equation can be simplified by utilizing the fact that its roots are bounded by the eigenvalues d_i of the matrix D. Indeed if we denote the roots as $\lambda_1, \lambda_2, \ldots, \lambda_n$ and assume that $\lambda_i \leq \lambda_{i+1}$ and $d_i \leq d_{i+1}$, it can be shown that

1. if $\sigma \geq 0$,

$$d_i \leq \lambda_i \leq d_{i+1}\,, \qquad i = 1, \ldots, n-1\,, \tag{24}$$

$$d_n \leq \lambda_n \leq d_n + \sigma \mathbf{u}^{\mathsf{T}}\mathbf{u}\,; \tag{25}$$

2. if $\sigma \leq 0$,

$$d_{i-1} \leq \lambda_i \leq d_i\,, \qquad i = 2, \ldots, n\,, \tag{26}$$

$$d_1 + \sigma \mathbf{u}^{\mathsf{T}}\mathbf{u} \leq \lambda_1 \leq d_1\,. \tag{27}$$

1.3.2 Symmetric row/column update

The matrix A to be diagonalized is given in the form

$$\mathrm{A} = \mathrm{D} + \mathbf{e}(p)\mathbf{u}^{\mathsf{T}} + \mathbf{u}\mathbf{e}(p)^{\mathsf{T}} = \begin{bmatrix} d_1 & \ldots & u_1 & \ldots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ u_1 & \ldots & d_p & \ldots & u_n \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \ldots & u_n & \ldots & d_n \end{bmatrix}\,, \tag{28}$$

where D is a diagonal matrix with diagonal elements $\{d_i | i = 1, \ldots, n\}$, $\mathbf{e}(p)$ is the unit vector in the p-direction, and $\mathbf{u}$ is a given update vector where the p-th element can be assumed to equal zero, $u_p = 0$, without loss of generality. Indeed, if the element is not zero, one can simply redefine $d_p \to d_p + 2u_p$, $u_p \to 0$.

The eigenvalue equation for matrix A is given as

$$(D - \lambda)\mathbf{x} + \mathbf{e}(p)\mathbf{u}^{\mathsf{T}}\mathbf{x} + \mathbf{u}\mathbf{e}(p)^{\mathsf{T}}\mathbf{x} = 0\,, \tag{29}$$

where $\mathbf{x}$ is an eigenvector and λ is the corresponding eigenvalue. The component number p of this vector-equation reads

$$(d_p - \lambda)x_p + \mathbf{u}^{\mathsf{T}}\mathbf{x} = 0\,, \tag{30}$$

while the component number $k \neq p$ reads

$$(d_k - \lambda)x_k + u_k x_p = 0\,, \tag{31}$$

Dividing the last equation by $(d_k - \lambda)$, multiplying from the left with $\sum_{k=1}^{n} u_k$, substituting $\mathbf{u}^{\mathsf{T}}\mathbf{x}$ using equation (30) and dividing by x_p gives the secular equation,

$$-(d_p - \lambda) + \sum_{k \neq p}^{n} \frac{u_k^2}{d_k - \lambda} = 0\,, \tag{32}$$

which determines the updated eigenvalues.

1.3.3 Symmetric rank-2 update

A symmetric rank-2 update can be represented as two consecutive rank-1 updates,

$$\mathbf{u}\mathbf{v}^\mathsf{T} + \mathbf{v}\mathbf{u}^\mathsf{T} = \mathbf{a}\mathbf{a}^\mathsf{T} - \mathbf{b}\mathbf{b}^\mathsf{T} \,, \tag{33}$$

where

$$\mathbf{a} = \frac{1}{\sqrt{2}}(\mathbf{u} + \mathbf{v}) \,, \; \mathbf{b} = \frac{1}{\sqrt{2}}(\mathbf{u} - \mathbf{v}) \,. \tag{34}$$

The eigenvalues can then be found by applying the rank-1 update method twice.

1.4 Singular Value Decomposition

Singular Value Decomposition (SVD) is a factorization of matrix A in the form

$$\mathrm{A} = \mathrm{U}\mathrm{S}\mathrm{V}^\mathsf{T} \,, \tag{35}$$

where S is a diagonal matrix, U is (generally) a semi-orthogonal matrix ($\mathrm{U}^\mathsf{T}\mathrm{U} = 1$), and V is an orthogonal matrix ($\mathrm{V}^\mathsf{T}\mathrm{V} = \mathrm{V}\mathrm{V}^\mathsf{T} = 1$).

The elements of the diagonal matrix S are called the *singular values* of matrix A. Singular values can always be chosen non-negative by changing the signs of the corresponding columns of matrix U. The columns of the matrices U and V are called the (correspondingly left and right) singular vectors.

1.4.1 Applications of SVD

Column space, null space, and rank The SVD of a matrix A provides a representation of the *column space* (also called *range*) and the *null space* of the matrix: the columns of the V matrix corresponding to vanishing singular values span the null space while the columns of the U matrix corresponding to non-vanishing singular values span the range of the matrix.

The rank of the matrix is given by the number of non-vanishing singular values.

Pseudoinverse An $n \times m$ matrix A has its *pseudoinverse* $m \times n$ matrix A^- if the following equation is satisfied,

$$\mathrm{A}\mathrm{A}^-\mathrm{A} = \mathrm{A} \,. \tag{36}$$

Given the SVD of the matrix, $\mathrm{A} = \mathrm{U}\mathrm{S}\mathrm{V}^\mathsf{T}$, its pseudoinverse is

$$\mathrm{A}^- = \mathrm{V}\mathrm{S}^-\mathrm{U}^\mathsf{T} \,, \tag{37}$$

where S^- is the pseudoinverse of the matrix S which is formed by replacing non-vanishing elements of S with their inverses.

Least squares solution The least sqaures solution to an overdetermined system $\mathrm{A}x = b$ is given via the matrix' pseudoinverse as

$$x = \mathrm{A}^- b \,. \tag{38}$$

Solution to homogenious systems Solutions to a homogenious system $\mathrm{A}x = 0$ are given by the null-space of the matrix, that is, by any linear combination of the columns of the matrix V corresponding to vanishing singular values of matrix A.

Lower-rank matrix approximation Sometimes one needs to approximate a matrix A with a smaller matrix $\tilde{\mathrm{A}}$ of a given rank r. The SVD of A provides one such approximation,

$$\tilde{\mathrm{A}} = \tilde{\mathrm{U}}\tilde{\mathrm{S}}\tilde{\mathrm{V}}^{\mathsf{T}} \,, \tag{39}$$

where $\tilde{\mathrm{S}}$ is a diagonal matrix which contains only the r largest singular values, and where $\tilde{\mathrm{U}}$ and $\tilde{\mathrm{V}}^{\mathsf{T}}$ contain only the corresponding singular vectors. It minimizes the *Frobenius norm* of the difference between the matrix and its approximation.

1.4.2 Relation to eigenvalues of $\mathrm{A}^{\mathsf{T}}\mathrm{A}$

Singular values are equal the square roots of the eigenvalues of the real symmetrix matrix $\mathrm{A}^{\mathsf{T}}\mathrm{A}$. Indeed, by construction the matrix $\mathrm{A}^{\mathsf{T}}\mathrm{A}$ is positive definite, therefore its eigenvalues are not-negiative and its eigenvalue decomposition can be written as

$$\mathrm{A}^{\mathsf{T}}\mathrm{A} = \mathrm{V}\mathrm{S}^2\mathrm{V}^{\mathsf{T}} \tag{40}$$

where S is the diagonal matrix of square roots of $\mathrm{A}^{\mathsf{T}}\mathrm{A}$, and V is the matrix of corresponding eigenvectors. Then the singular value decomposition of matrix A can be formally written as

$$\mathrm{A} = \mathrm{U}\mathrm{S}\mathrm{V}^{\mathsf{T}} \,, \; \mathrm{U} = \mathrm{A}\mathrm{V}\mathrm{S}^{-1} \,. \tag{41}$$

1.4.3 Calculation of SVD

While it is theoretically possible to calculate the SVD of a matrix A by diagonalizing $\mathrm{A}^{\mathsf{T}}\mathrm{A}$, it is not recommended in practice for several reasons including the potential numerical issues related to squaring small singular values leading to rounding errors. Generally it is more efficient to use algorithms specifically designed for SVD computations.

The most widely used algorithm, called Golub-Reinsch, is a generalization of the QR/QL algorithm for matrix diagonalization. However, like QR/QL in matrix diagonalization, it is a bit tedious and tricky to implement for a learner, therefore we shall discuss here two other, classical, methods which might not be as fast as Golub-Reinsch, but they are stable, precise, and easy to implement.

One-sided Jacobi SVD algorithm This is an iterative algorithm [1] where the given matrix A is gradually transformed into a matrix with orhogonal columns with the elementary iteration given as

$$\mathrm{A} \leftarrow \mathrm{A}\mathrm{J}(p, q, \theta) \,, \tag{42}$$

where the (rotation) angle θ is chosen such that the columns of the matrix A with the indices p and q become (after the rotation) orthogonal[2]. The indices (p, q) are swept cyclically, $(p = 1 \ldots m, q = p+1, \ldots, m)$, where m is the number of columns.

After the algoritm has converged, that is, the transformed matrix A has orthogonal columns, the SVD of the original matrix is recovered as follows: the matrix V is the accumulation of the rotation matrices J, the matrix U is given by normalizing the columns of the transformed matrix A, and the diagonal matrix S contains the norms of the columns of the transformed matrix A.

Two-sided Jacobi SVD algorithm This is also an iterative algorithm where a (square, else see below) matrix A is transformed ino a diagonal matrix, where in the elementary iteration one first applies a Givens rotation to symmetrize a pair of off-diagonal elements of the matrix and then applies a Jacobi transformation to eliminate these off-diagonal elements,

$$\mathrm{A} \to \mathrm{J}^\mathsf{T}\mathrm{G}^\mathsf{T}\mathrm{A}\mathrm{J}\,. \tag{44}$$

Just like in the Jacobi eigenvalue algorithm the iterations are performed in cyclic sweeps over all non-diagonal elements of the matrix.

For a 2×2 matrix the two-sided Jacobi SVD transformation is given as following: first, one applies a Givens rotation to symmetrize two off-diagonal elements,

$$\mathrm{A} = \begin{bmatrix} w & x \\ y & z \end{bmatrix} \to \mathrm{G}^\mathsf{T}\mathrm{A} = \begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} w & x \\ y & z \end{bmatrix} = \begin{bmatrix} a & b \\ b & c \end{bmatrix}\,, \tag{45}$$

where the rotation angle $\varphi = \mathrm{atan2}(x-y, w+z)$; and, second, one makes the usual Jacobi transformation to eliminate the off-diagonal elements b,

$$\begin{aligned} \mathrm{GA} &\to \mathrm{J}^\mathsf{T}\mathrm{G}^\mathsf{T}\mathrm{A}\mathrm{J} \\ &= \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} a & b \\ b & c \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} = \begin{bmatrix} d_1 & 0 \\ 0 & d_1 \end{bmatrix}. \end{aligned} \tag{46}$$

The matrices U and V are accumulated (from identity matrices) as

$$\mathrm{U} \leftarrow \mathrm{U}\mathrm{G}\mathrm{J}\,, \tag{47}$$

$$\mathrm{V} \leftarrow \mathrm{V}\mathrm{J}\,. \tag{48}$$

If the matrix A is a tall $n \times m$ non-square matrix $(n > m)$, the first step should be the QR-decomposition,

$$\mathrm{A} = \mathrm{Q}\mathrm{R}\,, \tag{49}$$

where Q is the $n \times m$ orthogonal matrix and R is a square triangular $m \times m$ matrix.

[2]The angle is given as

$$\theta = \frac{1}{2}\mathrm{atan2}(2\mathrm{A}_p^\mathsf{T}\mathrm{A}_q, \mathrm{A}_q^\mathsf{T}\mathrm{A}_q - \mathrm{A}_p^\mathsf{T}\mathrm{A}_p)\,, \tag{43}$$

where A_p and A_q are the corresponding columns of the matris A.

The second step is the normal SVD of the square matrix R,

$$\mathrm{R} = \mathrm{U}'\mathrm{D}\mathrm{V}^{\mathsf{T}} \,. \tag{50}$$

Now the SVD of the original matrix A is given as

$$\mathrm{A} = \mathrm{U}\mathrm{D}\mathrm{V}^{\mathsf{T}} \,, \tag{51}$$

where

$$\mathrm{U} = \mathrm{Q}\mathrm{U}' \,. \tag{52}$$

References

[1] P.P.M. de Rijk. A one-sided jacobi algorithm for computing the singular value decomposition on a vector computer. *SIAM J. Sci. Stat. Comput.*, 10(2):359, 1989.

[2] Gene H. Golub. Some modified matrix eigenvalue problems. *SIAM Rev.*, 15(2):318–334, 1973.