Eigenvalues

In addition to solving linear equations another important task of linear algebra is finding eigenvalues.

Let F be some operator and $\mathbf{x}$ a vector. If F does not change the direction of the vector $\mathbf{x}$, $\mathbf{x}$ is an *eigenvector* of the operator, satisfying the equation

$$F(\mathbf{x}) = \lambda \mathbf{x}, \tag{1}$$

where λ is a real or complex number, the *eigenvalue* corresponding to the eigenvector. Thus the operator will only change the length of the vector by a factor given by the eigenvalue.

If F is a linear operator, $F(a\mathbf{x}) = aF(\mathbf{x}) = a\lambda\mathbf{x}$, and hence $a\mathbf{x}$ is an eigenvector, too. Eigenvectors are not uniquely determined, since they can be multiplied by any constant.

In the following only eigenvalues of matrices are discussed. A matrix can be considered an operator mapping a vector to another one. For eigenvectors this mapping is a mere scaling.

Let $\mathbf{A}$ be a square matrix. If there is a real or complex number λ and a vector $\mathbf{x}$ such that

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}, \tag{2}$$

λ is an eigenvalue of the matrix and $\mathbf{x}$ an eigenvector. The quation (2) can also be written as

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{x} = 0.$$

If the equation is to have a nontrivial solution, we must have

$$\det(\mathbf{A} - \lambda\mathbf{I}) = 0. \tag{3}$$

When the determinant is expanded, we get the *characteristic polynomial*, the zeros of which are the eigenvalues.

If the matrix is symmetric and real valued, the eigenvalues are real. Otherwise at least some of them may be complex. Complex eigenvalues appear always as pairs of complex conjugates.

For example, the characteristic polynomial of the matrix

$$\begin{pmatrix} 1 & 4 \\ 1 & 1 \end{pmatrix}$$

is

$$\det \begin{pmatrix} 1-\lambda & 4 \\ 1 & 1-\lambda \end{pmatrix} = \lambda^2 - 2\lambda - 3.$$

The zeros of this are the eigenvalues $\lambda_1 = 3$ and $\lambda_2 = -1$.

Finding eigenvalues usign the characteristic polynomial is very laborious, if the matrix is big. Even determining the coefficients of the equation is difficult. The method is not suitable for numerical calculations.

To find eigenvalues the matrix must be transformed to a more suiteble form. Gaussian elimination would transform it into a triangular matrix, but unfortunately the eigenvalues are not conserved in the transform.

Another kind of transform, called a *similarity transform*, will not affect the eigenvalues. Similarity transforms have the form

$$\mathbf{A} \to \mathbf{A}' = \mathbf{S}^{-1}\mathbf{A}\mathbf{S},$$

where $\mathbf{S}$ is any nonsingular matrix, The matrices $\mathbf{A}$ and $\mathbf{A}'$ are called *similar*.

Power method

The power method is a simple method for finding the largest eigenvalue and the corresponding eigenvector of a matrix.

Basically the same idea as in solving an equation by direct iteration.

The algorithm proceeds as

$$\begin{aligned}
&\mathbf{x}_0 = \text{initial guess},\\
&\mathbf{y}_1 = A\mathbf{x}_0,\\
&\mathbf{x}_1 = \mathbf{y}_1 / \parallel \mathbf{y}_1 \parallel,\\
&\quad \ldots\\
&\mathbf{y}_k = A\mathbf{x}_{k-1},\\
&\mathbf{x}_k = \mathbf{y}_k / \parallel \mathbf{y}_k \parallel .
\end{aligned}$$

The iteration is terminated when the values do not change more than the required accuracy.

The eigenvector is then the last iterata $\mathbf{x}_k$ and the corresponding eigenvalue is

$$\| \mathbf{y}_k \| / \| \mathbf{x}_{k-1} \| .$$

```
program power
! find the largest eigenvalue by the power method
implicit none
 integer, parameter:: n=2
 real, dimension(n,n) :: a
 real, dimension(n):: x0, x1, y
 real :: norm, lambda, limit=0.0001
 integer :: i

! the matrix
a(1,:) = (/ 1, 4/)
a(2,:) = (/ 1, 1/)

! initial guess of the eigenvector
x0 = 1.0
```

```
! iterate until the eigenvector does not change
do
 do i=1,n
  y(i) = dot_product(a(i,:), x0)
 end do
 norm = sqrt(dot_product(y, y))
 x1 = y/norm
 write(*,*) x1
 if (maxval(abs(x1-x0)) < limit) then
  lambda =  sqrt(dot_product(y,y) / dot_product(x0, x0))
  exit
 end if
 x0 = x1
end do

write(*,'("eigenvalue = "F8.3)') lambda
write(*,'("eigenvector= "3F8.3)') x0

end program
```

```
>./a.out
  0.9284767    0.3713907
  0.8804711    0.4740998
  0.8987685    0.4384236
  0.8929464    0.4501630
  0.8949170    0.4462326
  0.8942635    0.4475408
  0.8944817    0.4471046
  0.8944089    0.4472499
  0.8944333    0.4472015
eigenvalue =   3.000
eigenvector=   0.894   0.447
```

The convergenc can be quite slow unless the highest eigenvalue is much bigger than the smaller ones.

QR decomposition

A commonly used method for finding eigenvalues is known as the QR method. The method is an iteration that repeatedly computes a decomposition of the matrix know as its QR decomposition. The decomposition is obtained in a finite number of steps, and it has some other uses, too. We'll first see how to compute this decomposition.

The QR decomposition of a matrix $\mathbf{A}$ is

$$\mathbf{A} = \mathbf{QR},$$

where $\mathbf{Q}$ is an orthogonal matrix and $\mathbf{R}$ an upper triangular matrix. This decomposition is possible for all matrices.

There are several methods for finding the decomposition

1) Householder transform

2) Givens rotations

3) Gram–Schmidt orthogonalisation

In the following we discuss the first two methods with exemples. They will probably be easier to understand than the formal algorithm.

Householder transform

The idea of the Householder transform is to find a set of transforms that will make all elements in one column below the diagonal vanish.

Assume that we have to decompose a matrix

$$\mathbf{A} = \begin{pmatrix} 3 & 2 & 1 \\ 1 & 1 & 2 \\ 2 & 1 & 3 \end{pmatrix}$$

We begin by taking the first column of this

$$\mathbf{x}_1 = \mathsf{a}(:, 1) = \begin{pmatrix} 3 \\ 1 \\ 2 \end{pmatrix}$$

and compute the vector

$$\mathbf{u}_1 = \mathbf{x}_1 - \parallel \mathbf{x}_1 \parallel \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} -0.7416574 \\ 1 \\ 2 \end{pmatrix}$$

This is used to create a Householder transformation matrix

$$\mathbf{P}_1 = \mathbf{I} - 2\frac{\mathbf{u}_1\mathbf{u}_1^T}{\| u_1 \|^2} = \begin{pmatrix} 0.8017837 & 0.2672612 & 0.5345225 \\ 0.2672612 & 0.6396433 & -0.7207135 \\ 0.5345225 & -0.7207135 & -0.4414270 \end{pmatrix}$$

It can be shown that this is an orthogonal matrix. It is easy to see this by calculating the scalar product of any two columns. The products are zeros, and thus the column vectors of the matrix are mutually orthogonal.

When the original matrix is multiplied by this transform the result is a matrix with zeros in the first column below the diagonal:

$$\mathbf{A}_1 = \mathbf{P}_1\mathbf{A} = \begin{pmatrix} 3.7416574 & 2.4053512 & 2.9398737 \\ 0. & 0.4534522 & -0.6155927 \\ 0. & -0.0930955 & -2.2311854 \end{pmatrix}$$

Then we use the second column to create a vector

$$\mathbf{x}_2 = \mathtt{a}(2:3,2) = \begin{pmatrix} 0.4534522 \\ -0.0930955 \end{pmatrix},$$

from which

$$\mathbf{u}_2 = \mathbf{x}_2 - \parallel \mathbf{x}_2 \parallel \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -0.0094578 \\ -0.0930955 \end{pmatrix}.$$

This will give the second transformation matrix

$$\mathbf{P}_2 = \mathbf{I} - 2\frac{\mathbf{u}_2\mathbf{u}_2^{\mathrm{T}}}{\parallel \mathbf{u}_2 \parallel^2} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.9795688 & -0.2011093 \\ 0 & -0.2011093 & -0.9795688 \end{pmatrix}$$

The product of $\mathbf{A}_1$ and the transformation matrix will be a matrix with zeros in the second column below the diagonal:

$$\mathbf{A}_2 = \mathbf{P}_2\mathbf{A}_1 = \begin{pmatrix} 3.7416574 & 2.4053512 & 2.9398737 \\ 0 & 0.4629100 & -0.1543033 \\ 0 & 0 & 2.3094011 \end{pmatrix}$$

Thus the matrix has been transformed to an upper triangular matrix. If the matrix is bigger, repeat the same procedure for each column until all the elements below the diagonal vanish.

Matrices of the decomposition are now obtained as

$$\mathbf{Q} = \mathbf{P}_1\mathbf{P}_2 = \begin{pmatrix} 0.8017837 & 0.1543033 & -0.5773503 \\ 0.2672612 & 0.7715167 & 0.5773503 \\ 0.5345225 & -0.6172134 & 0.5773503 \end{pmatrix}$$

$$\mathbf{R} = \mathbf{A}_2 = \mathbf{P}_2\mathbf{P}_1\mathbf{A} = \begin{pmatrix} 3.7416574 & 2.4053512 & 2.9398737 \\ 0 & 0.4629100 & -0.1543033 \\ 0 & 0 & 2.3094011 \end{pmatrix}.$$

The matrix $\mathbf{R}$ is in fact the $\mathbf{A}_k$ calculated in the last transformation; thus the original matrix $\mathbf{A}$ is not needed. If memory must be saved, each of the matrices $\mathbf{A}_i$ can be stored in the area of the previous one. Also, there is no need to keep the earlier matrices $\mathbf{P}_i$, but $\mathbf{P}_1$ will be used as the initial value of $\mathbf{Q}$, and at each step $\mathbf{Q}$ is always multiplied by the new tranformation matrix $\mathbf{P}_i$.

As a check, we can calculate the product of the factors of the decomposition to see that we will restore the original matrix:

$$\mathbf{QR} = \begin{pmatrix} 3 & 2 & 1 \\ 1 & 1 & 2 \\ 2 & 1 & 3 \end{pmatrix}.$$

Orthogonality of the matrix $\mathbf{Q}$ can be seen e.g. by calculating the product$\mathbf{Q}\mathbf{Q}^{\mathrm{T}}$:

$$\mathbf{Q}\mathbf{Q}^{\mathrm{T}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

In the general case the matrices of the decomposition are

$$\begin{aligned} \mathbf{Q} &= \mathbf{P}_1 \mathbf{P}_2 \cdots \mathbf{P}_n, \\ \mathbf{R} &= \mathbf{P}_n \cdots \mathbf{P}_2 \mathbf{P}_1 \mathbf{A}. \end{aligned}$$

Givens rotations

Another commonly used method is based on Givens rotation matrices:

$$P_{kl}(\theta) = \begin{pmatrix} 1 & & & & \\ & \cos\theta & \cdots & \sin\theta & \\ & \vdots & 1 & \vdots & \\ & -\sin\theta & \cdots & \cos\theta & \\ & & & & 1 \end{pmatrix}.$$

This is an orthogonal matrix.

Elements below the diagonal can be zeroed one at a time.

More efficient for sparse matrices (containing mostly zeroes).

Finding the eigenvalues

Eigenvalues can be found using iteratively the QR-algorithm, which will use the previous QR decomposition. If we started with the original matrix, the task would be computationally very time consuming. Therefore we start by transforming the matrix to a more suitable form.

A square matrix is in the block diagonal form if it is

$$\begin{pmatrix} T_{11} & T_{12} & T_{13} & \cdots & T_{1n} \\ 0 & T_{22} & T_{23} & \cdots & T_{2n} \\ 0 & 0 & \vdots & & \\ 0 & 0 & 0 & \cdots & T_{nn} \end{pmatrix},$$

where the submatrices T_{ij} are square matrices. It can be shown that the eigenvalues of such a matrix are the eigenvalues of the diagonal blocks T_{ii}.

If the matrix is a diagonal or triangular matrix, the eigenvalues are the diagonal elements. If such a form can be found, the problem is solved. Usually such a form cannot be obtained by a finite number of similarity transformations.

If the original matrix is symmetric, it can be transformed to a tridiagonal form without affecting its eigenvalues. In the case of a general matrix the result is a Hessenberg matrix, which has the form

$$\mathbf{H} = \begin{pmatrix} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \end{pmatrix}$$

The transformations required can be accomplished with Householder transforms or Givens rotations. The method is now slightly modified so that the elements immediately below the diagonal are not zeroed.

Transformation using the Householder transforms

As a first example, consider a symmetric matrix

$$\mathbf{A} = \begin{pmatrix} 4 & 3 & 2 & 1 \\ 3 & 4 & -1 & 2 \\ 2 & -1 & 1 & -2 \\ 1 & 2 & -2 & 2 \end{pmatrix}$$

We begin to transform this using Householder transform. Now we construct a vector $\mathbf{x}_1$ by taking only the elements of the first column that are below the diagonal:

$$\mathbf{x}_1 = \begin{pmatrix} 3 \\ 2 \\ 1 \end{pmatrix}$$

Using these form the vector

$$\mathbf{u}_1 = \mathbf{x}_1 - \parallel \mathbf{x}_1 \parallel \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} -0.7416574 \\ 2 \\ 1 \end{pmatrix}$$

and from this the matrix

$$\mathbf{p}_1 = \mathbf{I} - 2\mathbf{u}_1\mathbf{u}_1^{\mathrm{T}} / \parallel \mathbf{u}_1 \parallel = \begin{pmatrix} 0.8017837 & 0.5345225 & 0.2672612 \\ 0.5345225 & -0.4414270 & -0.7207135 \\ 0.2672612 & -0.7207135 & 0.6396433 \end{pmatrix}$$

and finally the Householder transformation matrix

$$\mathbf{P}_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0.8017837 & 0.5345225 & 0.2672612 \\ 0 & 0.5345225 & -0.4414270 & -0.7207135 \\ 0 & 0.2672612 & -0.7207135 & 0.6396433 \end{pmatrix}$$

Now we can make the similarity transform of the matrix $\mathbf{A}$. The transformation matrix is symmetric, so there is no need for transposing it:

$$\mathbf{A}_1 = \mathbf{P}_1\mathbf{A}\mathbf{P}_1 = \begin{pmatrix} 4 & 3.7416574 & 0 & 0 \\ 3.7416574 & 2.4285714 & 1.2977396 & 2.1188066 \\ 0 & 1.2977396 & 0.0349563 & 0.2952113 \\ 0 & 2.1188066 & 0.2952113 & 4.5364723 \end{pmatrix}$$

The second column is handled in the same way. First we form the vector $\mathbf{x}_2$

$$\mathbf{x}_2 = \begin{pmatrix} 1.2977396 \\ 2.1188066 \end{pmatrix}$$

and from this

$$\mathbf{u}_2 = \mathbf{x}_2 - \parallel \mathbf{x}_2 \parallel (1,0)^{\mathrm{T}} = \begin{pmatrix} -1.1869072 \\ 2.1188066 \end{pmatrix}$$

and

$$\mathbf{p}_2 = \begin{pmatrix} 0.5223034 & 0.8527597 \\ 0.8527597 & -0.5223034 \end{pmatrix}$$

and the final transformation matrix

$$\mathbf{P}_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 00 & \\ 0 & 0 & 0.5223034 & 0.8527597 \\ 0 & 0 & 0.8527597 & -0.5223034 \end{pmatrix}$$

Making the transform we get

$$\mathbf{A}_2 = \mathbf{P}_2\mathbf{A}_1\mathbf{P}_2 = \begin{pmatrix} 4 & 3.7416574 & 0 & 0 \\ 3.7416574 & 2.4285714 & 2.4846467 & 0 \\ 0 & 2.4846467 & 3.5714286 & -1.8708287 \\ 0 & 0 & -1.8708287 & 1 \end{pmatrix}$$

Thus we obtained a tridiagonal matrix, as we should in the case of a symmetric matrix.

As another example, we take an asymmetric matrix:

$$\mathbf{A} = \begin{pmatrix} 4 & 2 & 3 & 1 \\ 3 & 4 & -2 & 1 \\ 2 & -1 & 1 & -2 \\ 1 & 2 & -2 & 2 \end{pmatrix}$$

The transformation proceeds as before, and the result is

$$\mathbf{A}_2 = \begin{pmatrix} 4 & 3.4743961 & -1.3039935 & -0.4776738 \\ 3.7416574 & 1.7857143 & 2.9123481 & 1.1216168 \\ 0 & 2.0934787 & 4.2422252 & -1.0307759 \\ 0 & 0. & -1.2980371 & 0.9720605 \end{pmatrix},$$

which is of the Hessenberg form.

QR-algorithm

We now have a simpler tridiagonal or Hessenberg matrix, which still has the same eigenvalues as the original matrix. Let this transformed matrix be $\mathbf{H}$. Then we can begin to search for the eigenvalues. This is done by iteration.

As an initial value, take $\mathbf{A}_1 = \mathbf{H}$. Then repeat the following steps:

– Find the QR decomposition $\mathbf{Q}_i\mathbf{R}_i = \mathbf{A}_i$.

– Calculate a new matrix $\mathbf{A}_{i+1} = \mathbf{R}_i\mathbf{Q}_i$.

The matrix $\mathbf{Q}$ of the QR decomposition is orthogonal and $\mathbf{A} = \mathbf{QR}$, and so $\mathbf{R} = \mathbf{Q}^{\mathrm{T}}\mathbf{A}$. Therefore

$$\mathbf{RQ} = \mathbf{Q}^{\mathrm{T}}\mathbf{AQ}$$

is a similarity transform that will conserve the eigenvalues.

The sequence of matrices $\mathbf{A}_i$ converges towards an upper tridiagonal or block matrix, from which the eigenvalues can be picked up.

In the last example the limiting matrix after 50 iterations is

$$\begin{pmatrix} 7.0363389 & -0.7523758 & -0.7356716 & -0.3802631 \\ 4.265E-08 & 4.9650342 & -0.8892339 & -0.4538061 \\ 0 & 0 & -1.9732687 & -1.3234202 \\ 0 & 0 & 0 & 0.9718955 \end{pmatrix}$$

The diagonal elements are now the eigenvalues. If the eigenvalues are complex numbers, the matrix is a block matrix, and the eigenvalues are the eigenvalues of the diagonal 2×2 submatrices.

Singular value decomposition (SVD)

Eigenvalue programs can be used to find the SVD of a matrix.

The SVD of an $n \times m, n \geq m$ matrix $\mathbf{A}$ is

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T,$$

where $\mathbf{U}$ is an $n \times n$ orthogonal matrix, $\mathbf{V}$ is an $m \times m$ orthogonal matrix, and

$$\Sigma = \begin{pmatrix} \sigma_1 & & \\ & \vdots & \\ & & \sigma_m \\ 0 & & \end{pmatrix}$$

The values σ_i are called the *singular values* of the matrix $\mathbf{A}$. They are square roots of the eigenvalues of $\mathbf{A}^T\mathbf{A}$. They are arranged in descending order: $\sigma_1 \geq \sigma_2 \geq \ldots \sigma_n \geq 0$.

The largest singular value σ_1 equals the L_2 norm of the matrix.

The columns of $\mathbf{U}$ are the eigenvectors of $\mathbf{A}\mathbf{A}^T$.

The columns of $\mathbf{V}$ are the eigenvectors of $\mathbf{A}^T\mathbf{A}$.

Applications of the SVD:

Linear least squares fit (can be used also when the normal equations are singular or nearlyt singular; discussed later)

Data compression (also discussed later, maybe)