nag®

CSC / NAG Autumn School on

Core Algorithms in High-Performance Scientific Computing

Maths V

Dwight Barkley

Eigenvalue Problems II

Maths V: Iterative Methods for Eigenvalue Problems

5.1 Three Basic Iterative Methods for Eigenvalues

Here we consider three elementary iterative methods for finding an eigenvalue of a matrix A.

We assume A is non-defective and for simplicity we exclude the case of a real matrix A with complex eigenvalues. This is not a fundamental restriction, but the algorithms need modification to take into account the possibility of 2×2 blocks or equivalently complex output from real input. (Recall Maths lecture III.)

Rayleigh quotient

The **Rayleigh quotient** of a vector v is the scalar

$$r(v) = \frac{\langle v, Av \rangle}{\langle v, v \rangle}$$

If v is an eigenvector of A and λ is the corresponding eigenvalue, then $r(v) = \lambda$. If v is not an eigenvector of A then $r(v)$ can be thought of as the number most like the eigenvalue of v. $r(v)$ is a solution to

$$\min_r ||Av - rv||_2$$

The Rayleigh Quotient (RQ) has the following property. Let $v^{(k)}$ be an approximation to eigenvector v of A, then

$$|r(v^{(k)}) - r(v)| = O(||v^{(k)} - v||_2^2)$$

Eigenvalues determined by the RQ converge faster than eigenvectors.

For the remainder assume that the eigenvalues λ_j of A are ordered

$$|\lambda_1| \geq |\lambda_2| \geq \cdots |\lambda_j| \geq \cdots \geq |\lambda_n| \geq 0.$$

with corresponding eigenvectors $v_1, v_2, \cdots v_j, \cdots v_n$ which can be taken to be normalized $||v_j|| = 1$.

Power method

Method for finding the **dominant eigenvalue** (eigenvalue of largest magnitude) of A.

Algorithm (Power Method)

Choose $v^{(0)}$ with $||v^{(0)}|| = 1$.
For $k = 1, 2, 3$
$u = Av^{(k-1)}$
$v^{(k)} = u/||u||$
$\lambda^{(k)} = r(v^{(k)})$

In this and subsequent algorithms we strive to present the basic iteration. Stopping tests are omitted. One would almost surely not implement power method this way since hidden in $r(v)$ is an action of A and in practice each iteration requires only one action of A. Nevertheless power method is extremely easy to implement.

The reason that the power method will converge to the dominant eigenpair is as follows. The initial vector $v^{(0)}$ can be expressed as a linear superposition of the eigenvectors v_j of A:

$$v^{(0)} = \sum_j \alpha_j v_j$$

acting with A then gives

$$Av^{(0)} = A\sum_j \alpha_j v_j = \sum_j \alpha_j A v_j = \sum_j \lambda_j \alpha_j v_j$$

Thus, with every action of A, each component in the sum gets multiplied by the corresponding eigenvalue. After k actions of A (without normalization) we have

$$A^k v^{(0)} = A^k \sum_j \alpha_j v_j = \sum_j \alpha_j A^k v_j = \sum_j (\lambda_j)^k \alpha_j v_j$$

This sum will be dominated by the term corresponding to the dominant eigenvalue, which by our ordering is λ_1. Upon normalization we have

$$v^{(k)} = \frac{A^k v^{(0)}}{||A^k v^{(0)}||} \simeq \frac{1}{|\alpha_1 \lambda_1^k|} \sum_j \lambda_j^k \alpha_j v_j = \sigma^{(k)} v_1 + \left(\frac{\lambda_2}{|\lambda_1|}\right)^k \frac{\alpha_2}{|\alpha_1|} v_2 + \cdots$$

where $|\sigma^{(k)}| = 1$.

One can see that $v^{(k)}$ differs (up to a scalar) from v_1 by an error that is dominated by the ratio of λ_2/λ_1. It is not hard to prove that

$$||v^{(k)} - \sigma^{(k)} v_1||_2 = O(\left|\frac{\lambda_2}{\lambda_1}\right|^k)$$

and hence that

$$|\lambda^{(k)} - \lambda_1|_2 = O(\left|\frac{\lambda_2}{\lambda_1}\right|^{2k})$$

Note how the convergence rate is determined by the spectral properties of A. In this case by a certain ratio of eigenvalues of A.

The problem with power method is that it only finds dominant eigenvalues, and generally these are not the ones needed.

Inverse iteration

One can use a method similar in spirit to the power method to find the eigenvalue closest to any scalar s, (where s must be real if A is for the usual reason). The idea is to iterate the matrix

$$(A - sI)^{-1}$$

rather than A. s is called a **shift** and this method is sometimes called **shift-and-invert**.

$(A - sI)^{-1}$ and A have the same eigenvectors but the eigenvalues are related via

$$\mu_j = \frac{1}{\lambda_j - s}$$

Clearly $|\mu_j|$ will be largest for $|\lambda_j - s|$ smallest and hence power iteration on $(A - sI)^{-1}$ will converge to the eigenvector of A whose eigenvalue is closest to s.

Algorithm (Inverse Iteration)

Choose $v^{(0)}$ with $||v^{(0)}|| = 1$.
For $k = 1, 2, 3$
  Solve $(A - sI)u = v^{(k-1)}$ for u
  $v^{(k)} = u/||u||$
  $\lambda^{(k)} = r(v^{(k)})$

It is easy to adapt the previous convergence rates to this method. If one has a good guess for the desired eigenvalue and no other eigenvalues are closer, then the method will generally converge very quickly. The disadvantage of the method is the need to solve a linear system of equations each iterations. However, the matrix $(A - sI)$ is fixed for all iterations, and hence if an LU decomposition is feasible, each iteration can be done cheaply.

Rayleigh quotient iteration

Since the convergence of inverse iteration is set by the closeness of the shift s to the desired eigenvalue λ_j, convergence can be sped up by changing the shift each iteration to best the last best guess based on the Rayleigh quotient. In effect we consider a sequence of matrices

$$(A - s^{(k)}I)^{-1}$$

where $s^{(k)} = \lambda^{(k-1)}$.

Algorithm (Rayleigh Quotient Iteration)

Choose $v^{(0)}$ with $||v^{(0)}|| = 1$. Set $\lambda^{(0)} = r(v^{(0)})$
For $k = 1, 2, 3$
  Solve $(A - \lambda^{(k-1)}I)u = v^{(k-1)}$ for u
  $v^{(k)} = u/||u||$
  $\lambda^{(k)} = r(v^{(k)})$

The advantage of RQ iteration is that the convergence is ultimately extremely fast – one of the fastest methods you will ever see. If λ_j is the actual eigenvalue of A to which the method is converging, then the following holds

$$|\lambda^{(k)} - \lambda_j| = O(|\lambda^{(k-1)} - \lambda_j|^3)$$

This is known as cubic convergence since the error at iteration $e_k = |\lambda^{(k)} - \lambda_j|$ is the cube (up to a constant) of the error at iteration $k - 1$.

The primary disadvantage of RQ iteration is that since the matrix $(A - \lambda^{(k-1)}I)$ changes each iteration, it is generally not competitive with inverse iteration if a direct method is used for the linear solve.

A further disadvantage of RQ iteration is that even with a reasonable initial guess for the eigenvalue of interest, the method might wander off initially and hence converge to an eigenvalue other than the desired one. It might be necessary to perform a few inverse iterations first before turning to RQ iteration.

5.2 Rational maps - the Cayley transform

We now generalize inverse iteration. The following is the crucial observation. **Let g be a rational function (rational map). Then if v is an eigenvector of A with eigenvalue λ then v is also an eigenvector of $g(A)$ with eigenvalue $g(\lambda)$. We denote the eigenvalue of $g(A)$ by μ. The eigenvalues are mapped, but not the eigenvectors.**

For example, inverse iteration corresponds to

$$g(A) = \frac{1}{A - sI}$$

where the denominator is understood to mean matrix inverse. Thereby g gives a mapping of the eigenvalues:

$$g : \lambda \to \mu = \frac{1}{\lambda - s}$$

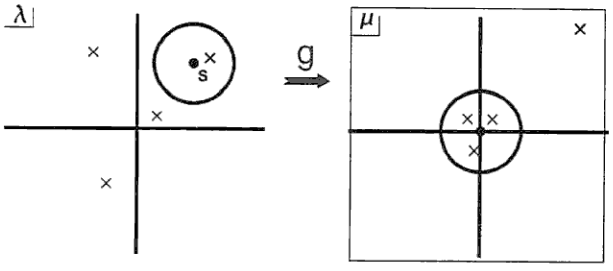

Figure 5.1: Mapping of the eigenvalue spectrum in inverse iteration: $g(A) = 1/(A - sI)$.

The effect of such a mapping is shown. Eigenvalues λ near s are mapped to eigenvalues μ with large magnitude. Eigenvalues far from s are mapped to near the origin. Inverse iteration is just power iteration performed on the mapped system. The mapping has two beneficial effect. First is allows one to select, via the choice of s the eigenvalue λ to target and second it can greatly enhance the magnitude of the mapped eigenvalue and hence the convergence rate.

One can see that there is no need to restrict simply to inverse iteration. A very common mapping used in practice is the Cayley transform

$$g(A) = \frac{A - tI}{A - sI}.$$

A variety of other notations are used for such rational functions.

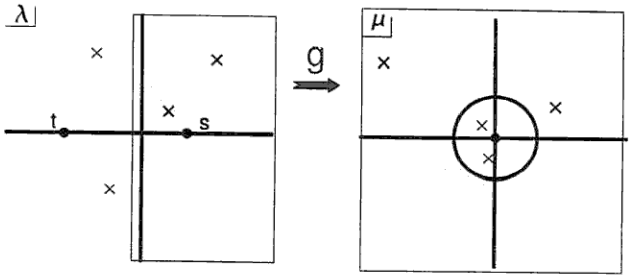

Figure 5.2: Mapping of the eigenvalue spectrum by the Cayley transform.

5.3 Simultaneous Iteration (block power method)

The methods just considered compute only one eigenvalue (or one eigenvalue at time). One frequently wants multiple eigenvalues. A basic method for finding a few eigenvalues is to simultaneously iterate K vectors. This also, allow one to find complex eigenvalues of a real matrix.

Let

$$V^{(0)} = \left(v_1^{(0)} \;\middle|\; v_2^{(0)} \;\middle|\; \cdots \;\middle|\; v_m^{(0)} \right)$$

be an $n \times m$ matrix whose columns are m vectors to be iterated. Then

$$AV^{(0)} = \left(Av_1^{(0)} \;\middle|\; Av_2^{(0)} \;\middle|\; \cdots \;\middle|\; Av_m^{(0)} \right)$$

Repeating gives a **simultaneous** iteration of m vectors. If one proceeds with such iterations, then after some number of iterations the columns of $V^{(k)}$ will all approach the same vector, namely v_1 corresponding to the dominant eigenvalue of A. In order to prevent this, one orthonormalizes the m vectors after each iteration using an algorithm called Gram-Schmidt orthonormalization. For the purposes here, the result is $\hat{Q}\hat{R}$ where $\hat{Q}$ is an $n \times m$ matrix whose columns are orthonormal ($\hat{Q}^*\hat{Q} = I$) and $\hat{R}$ is an $m \times m$ upper triangular matrix.

Algorithm (Simultaneous Iteration or Power Method)

Choose $V^{(0)}$ with m orthonormal columns.
For $k = 1, 2, 3$
  $W = AV^{(k-1)}$
  Perform modified GS to obtain $W = \hat{Q}\hat{R}$

$$V^{(k)} = \hat{Q}$$

The iterations will lead to $V^{(k)}$ spanning the space of m dominant eigenvectors of A. More specifically, consider sufficiently many iterations such that $V^{(k)} \simeq V^{(k-1)}$ to some desired precision. Looking at the algorithm we see that at each step:

$$AV^{(k-1)} = V^{(k)}\hat{R}$$

Letting $V = V^{(k)} \simeq V^{(k-1)}$ be the converged $n \times m$ matrix. We have

$$AV = V\hat{R}$$

or

$$V^T AV = \hat{R}$$

Thus we have a (reduced) Schur decomposition of A corresponding to the m dominant eigenvalues. The eigenvalues can be read off from the diagonal of $\hat{R}$.

Following the discussion of rational maps, simultaneous iteration may be applied not only to A but some rational mapping of A.

Simultaneous iteration is generally frowned upon in comparison with the Arnoldi method discussed next. It is more costly than Arnoldi but it has an important advantage in practice: it will converge and give you the m eigenvalues you want. This is not always the case with Arnoldi. In practice if one should consider using m slightly larger than the number of desired eigenvalues.

At final note: in the case $m = n$ (where A is $n \times n$) and $V^{(0)} = I$ this becomes what is known as the QR method.

5.4 Krylov Subspaces and Arnoldi's Method

Before considering Arnoldi's methods, we first motivate it in two ways.

First way

Recall that when considering direct methods for eigenvalues, an important step (the expensive step), was the transformation to upper Hessenberg form

$$Q^* AQ = H$$

Write this as

$$AQ = QH$$

Pictorially

$$\begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{pmatrix} \begin{pmatrix} & & & & \\ q_1 & q_2 & q_3 & q_4 & q_5 \\ & & & & \end{pmatrix} = \begin{pmatrix} & & & & \\ q_1 & q_2 & q_3 & q_4 & q_5 \\ & & & & \end{pmatrix} \begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & & \times & \times \end{pmatrix}$$

If you try to get an exact expression using just the upper corner of H and this is what you will find.

Cutting the right-most column from the expression gives:

$$\begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{pmatrix} \begin{pmatrix} & & & \\ q_1 & q_2 & q_3 & q_4 \\ & & & \end{pmatrix} = \begin{pmatrix} & & & & \\ q_1 & q_2 & q_3 & q_4 & q_5 \\ & & & & \end{pmatrix} \begin{pmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ & \times & \times & \times \\ & & \times & \times \\ & & & \times \end{pmatrix}$$

The Q matrix on the left and the H matrix on the right are now rectangular matrices. Now cut the next column.

$$\begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{pmatrix} \begin{pmatrix} & & \\ q_1 & q_2 & q_3 \\ & & \end{pmatrix} = \begin{pmatrix} & & & & \\ q_1 & q_2 & q_3 & q_4 & q_5 \\ & & & & \end{pmatrix} \begin{pmatrix} \times & \times & \times \\ \times & \times & \times \\ & \times & \times \\ & & \times \\ & & \end{pmatrix}$$

Now if you look on the right you will see that the last column q_5 multiplies the bottom row of the rectangular H matrix which is now all zeros. Hence, this column (which is sometimes called silent) can be removed giving

$$\begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{pmatrix} \begin{pmatrix} & & \\ q_1 & q_2 & q_3 \\ & & \end{pmatrix} = \begin{pmatrix} & & & \\ q_1 & q_2 & q_3 & q_4 \\ & & & \end{pmatrix} \begin{pmatrix} \times & \times & \times \\ \times & \times & \times \\ & \times & \times \\ & & \times \end{pmatrix}$$

One can continue cutting the right most column from both sides. Each time one does, not only does one remove a column from H, but because H is upper Hessenberg, one can also remove the bottom row and a column from the Q matrix on the right-hand-side.

$$\begin{pmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{pmatrix} \begin{pmatrix} & \\ q_1 & q_2 \\ & \end{pmatrix} = \begin{pmatrix} & & \\ q_1 & q_2 & q_3 \\ & & \end{pmatrix} \begin{pmatrix} \times & \times \\ \times & \times \\ & \times \end{pmatrix}$$

Back to symbols, we have

$$AQ_k = Q_{k+1}\tilde{H}_k$$

where Q_k means an $n \times k$ matrix with orthogonal columns. $\tilde{H}_k$ is an upper Hessenberg, $(k+1) \times k$ matrix. It is standard notation in numerical linear algebra to use $\tilde{H}_k$ or $\hat{H}_k$ to indicate that a matrix is rectangular. We shall use this for H but not Q_k.

Points to note. The expression is exact. The Q matrix on the right is the same as on the left except that it has one more column. The H matrix is rectangular, upper Hessenberg. One is interested in the case were $k \ll n$.

The point of this little exercise is to show the structure of the matrices we will obtain by Arnoldi's method. One certainly would not compute the expression this way.

Second way

This way will be equivalent to Arnoldi's method in exact arithmetic. Consider a Krylov sequence. We start from a vector which we continue to call b, but it is now arbitrary. Generate a sequence of k vectors by $k-1$ actions of A

$$\{b, Ab, A^2b, \cdots, A^{k-1}b\}$$

Again we have a Krylov subspace $\mathcal{K}$ spanned by these k vectors. For convenience of notation we call these vectors $v_1, \ldots, v_k$

$$\{b, Ab, A^2b, \cdots, A^{k-1}b\} = \{v_1, v_2, \cdots, v_k\}$$

We now put these vectors into the columns of a matrix.

$$V_k = \begin{pmatrix} & & & \\ v_1 & v_2 & \ldots & v_k \\ & & & \end{pmatrix}$$

The is called a **Krylov matrix**. Despite the appearance here, V_k is very tall and skinny in practice since $k \ll n$. There is likewise a similar matrix V_{k+1} containing one more Krylov vector.

$$V_{k+1} = \begin{pmatrix} & & & & \\ v_1 & v_2 & \ldots & v_k & v_{k+1} \\ & & & & \end{pmatrix}$$

As we saw before, due to the way Krylov vectors are constructed, there is a simple relationship between AV_k and V_{k+1}.

$$\overbrace{\underbrace{\{v_1, v_2, v_3, \cdots, v_k, v_{k+1}\}}_{V_k}}^{AV_k}$$

As a matrix equation this is:

$$AV_k = V_{k+1}\tilde{S}_k$$

where $\tilde{S}_k$ is a $(k+1) \times k$ matrix with 1's down the first sub-diagonal and zero elsewhere. $\tilde{S}_k$ is not symmetric. It is in fact upper Hessenberg and corresponds to a shift by one though the Krylov sequence.

Now if one generates by some means a (reduced) QR decomposition of V_k and V_{k+1} this can be written

$$AQ_kR_k = Q_{k+1}R_{k+1}\tilde{S}_k$$

where Q_k is $n \times k$ with orthogonal columns and R_k is upper triangular. Similarly with Q_{k+1} and R_{k+1}. Again because of the Krylov structure, Q_k and Q_{k+1} agree in the first k columns. Moving R_k to the right side gives:

$$AQ_k = Q_{k+1}\, R_{k+1}\tilde{S}_k(R_k)^{-1}$$

Defining

$$\tilde{H}_k = R_{k+1}\tilde{S}_k(R_k)^{-1}$$

we have finally

$$AQ_k = Q_{k+1}\tilde{H}_{k+1}$$

The matrix $\tilde{H}_{k+1}$ is in fact $(k+1) \times k$ upper Hessenberg. It inherits this from directly from the $\tilde{S}_k$ matrix and is a direct consequence of the fact that the initial matrices were Krylov matrices.

The final expression is precisely the result that is obtained by Arnoldi method. For numerical stability, the calculations are order in a different way that just given, but in exact arithmetic the result is the same Q matrices and upper Hessenberg $\tilde{H}_k$.

Because the expression relating the large A matrix to a small $\tilde{H}$ matrix was obtained by iterating on A, i.e. generating a Krylov sequence, one has a practical method for obtaining the Q's and $\tilde{H}$ in practice. Moreover, as one would expect based on the previous discussion of power method, that as we iterate, the Krylov spaces will contain the dominant eigenvectors of A. The small $(k+1) \times k$ matrix $\tilde{H}_k$ will contain all necessary information to obtain the dominant eigenvalues of A. It, together with the matrix Q_k, contains all information needed to obtain the eigenvectors of A.

One final manipulation is to separate the bottom row of right-hand-side

$$AQ_k = Q_{k+1}\tilde{H}_k = Q_k H_k + e$$

where H_k is now a square upper Hessenberg matrix. e represents $q_{k+1}H_{k+1,k}$ and will effectively encode the error in the eigenvalue approximation.

Arnoldi method

The algorithm for Arnoldi iteration is the following

Algorithm (Arnoldi Iteration)

Choose b arbitrary non-zero, $q_1 = b/||b||$.
For $k = 1, 2, 3, k-1$
  $v = Aq_k$
  for $j = 1$ to k
    $h_{jk} = \langle q_j, w \rangle$
    $w = w - h_{jk}q_j$
  $h_{k+1,k} = ||w||$
  $q_{k+1} = w/h_{k+1,k}$

The vectors q_k are the columns of the Q matrices and the quantities h_{jk} are the elements of H. Effectively one is performing QR decomposition on the Krylov vectors as they are generated by action of A. The result is we have a readily computable way to obtain

$$AQ_k = Q_k H_k + e$$

The final eigenvalue estimates for the eigenvalues of A are obtained by finding the eigenvalues of H_k by some direct method such a Schur decomposition. Since H_k is small and already upper Hessenberg, this is a trivial computation.

The (big) eigenvectors of A are obtained from the (small) eigenvectors of H_k as follows. Let w be an eigenvector of H_k with eigenvalue λ. Then

$$\begin{aligned} AQ_k w &= Q_k H_k w + ew \\ AQ_k w &= Q_k \lambda w + ew \\ AQ_k w &= \lambda Q_k w + ew \end{aligned}$$

Hence,

$$v = Q_k w$$

is an eigenvector of A with eigenvalue λ up to an exactly computable error ew:

$$Av = \lambda v + ew$$

In summary, the Arnoldi method for obtaining eigenvalues is:

1. Compute k steps of Arnoldi iteration.
2. Using a dense method, compute the eigenvalues and eigenvectors of the resulting upper Hessenberg matrix.
3. Compute the error ew corresponding to each eigenpair of interest. If converged, output all λ and $v = Q_k w$ of interest, otherwise return to step 1 and perform another iteration.

Most modern methods for obtaining eigenvalues for very large matrices are based on some form of Arnoldi iteration.