4.8 Arnoldi Iteration, Krylov Subspaces and GMRES

We start with the problem of using a similarity transformation to convert an $n \times n$ matrix A to upper Hessenberg form H, i.e.,

$$A = QHQ^*, \tag{30}$$

with an appropriate unitary matrix Q (see Section 5.7). Earlier we used Householder transformations to accomplish this goal because of their superior stability properties. Now we will use so-called *Arnoldi iteration*. This is analogous to using the modified Gram-Schmidt method instead of Householder's QR factorization. Householder factorization is more stable, but MGS has the advantage that the QR factorization of A is determined one column at a time. So that for a modified A (e.g., when a column is added to A) the factorization can be computed without having to start all over. Therefore, the advantage of MGS as well as Arnoldi iteration is that we can stop at any time, and then have a partial factorization.

Remark: Since this approach starts with a similarity transformation (30) Arnoldi iteration can also be applied to eigenvalue problems. However, we will concentrate on linear systems.

Note that (30) is equivalent to

$$AQ = QH$$

with $n \times n$ matrices A, Q, and H. If we take $m < n$ then we can rewrite this as

$$A \begin{bmatrix} q_1 & q_2 & \dots & q_m & q_{m+1} & \dots & q_n \end{bmatrix} = \begin{bmatrix} q_1 & q_2 & \dots & q_m & q_{m+1} & \dots & q_n \end{bmatrix} \times$$
$$\times \begin{bmatrix} h_{11} & h_{12} & & \dots & h_{1m} & & \dots & h_{1n} \\ h_{21} & h_{22} & & & h_{2m} & & \dots & h_{2n} \\ 0 & h_{32} & h_{33} & & h_{3m} & & \dots & h_{3n} \\ \vdots & \ddots & \ddots & & \vdots & & & \vdots \\ & & 0 & h_{m,m-1} & h_{mm} & & & \\ & & & 0 & h_{m+1,m} & \ddots & & \\ \vdots & & & & \ddots & \ddots & \ddots & \vdots \\ 0 & & \dots & & & 0 & h_{n,n-1} & h_{nn} \end{bmatrix},$$

where q_j, $j = 1, \dots, n$, are the columns of Q. Now we consider only part of this equation, namely

$$A \begin{bmatrix} q_1 & q_2 & \dots & q_m \end{bmatrix} = \begin{bmatrix} q_1 & q_2 & \dots & q_m & q_{m+1} \end{bmatrix} \begin{bmatrix} h_{11} & h_{12} & & \dots & h_{1m} \\ h_{21} & h_{22} & & & h_{2m} \\ 0 & h_{32} & h_{33} & & h_{3m} \\ \vdots & & \ddots & \ddots & \vdots \\ & & 0 & h_{m,m-1} & h_{mm} \\ 0 & \dots & & 0 & h_{m+1,m} \end{bmatrix},$$

or

$$AQ_m = Q_{m+1}\widetilde{H}_m,$$

where A is $n \times n$, Q_m is $n \times m$, Q_{m+1} is $n \times (m+1)$, and $\widetilde{H}_m$ is $(m+1) \times m$.

Comparing the m-th columns of the two sides we get

$$Aq_m = h_{1m}q_1 + h_{2m}q_2 + \ldots + h_{mm}q_m + h_{m+1,m}q_{m+1},$$

which we read as an $(m+1)$-term recurrence relation for q_{m+1}, i.e.,

$$q_{m+1} = \frac{Aq_m - \sum\limits_{j=1}^{m} h_{jm}q_j}{h_{m+1,m}}. \tag{31}$$

<u>Remark:</u> The recurrence formula (31) is analogous to the formula for the general step of the Gram-Schmidt method (see Section 5.3)

$$q_j = \frac{a_j - \sum\limits_{i=1}^{j-1} r_{ij}q_i}{r_{jj}}.$$

Arnoldi iteration has to be started with a (normalized) initial vector q_1. Then (31) reads

$$q_2 = \frac{Aq_1 - h_{11}q_1}{h_{21}}.$$

Since A and q_1 are given, we can compute the matrix-vector product Aq_1. Moreover, since we want q_2 to be orthogonal to q_1, we want

$$q_1^* q_2 = 0,$$

i.e.,

$$0 = q_1^* A q_1 - h_{11} \underbrace{q_1^* q_1}_{=1},$$

so that

$$h_{11} = q_1^* A q_1.$$

h_{21} is then used to normalize q_2, i.e.,

$$h_{21} = \|Aq_1 - h_{11}q_1\|.$$

The algorithm for Arnoldi iteration is analogous to that for the modified Gram-Schmidt algorithm of Section 5.3:

Algorithm (Arnoldi Iteration)

```
Input $A$, $b$
$q_1 = b/\|b\|$
for $m = 1, 2, \ldots$ do
    $v = Aq_m$
```

```
        for $j = 1$ to $m$ do
            $h_{jm} = q_j^* v$
            $v = v - h_{jm} q_j$
        end
        $h_{m+1,m} = \|v\|$
        $q_{m+1} = v / h_{m+1,m}$
    end
    Output $Q$, and nonzero part of $H$
```

Remarks:

1. The most expensive part of the algorithm is the computation of the matrix-vector product Aq_m that needs to be computed once every iteration. Therefore, for sparse matrices this can be done very efficiently. In fact, in a real implementation of the algorithm we don't even have to know the matrix A explicitly, only its action Aq which can be supplied in a separate subroutine.

2. Given an $n \times n$ matrix A and a vector $b \in \mathbb{C}^n$ we can generate a sequence of vectors $\{b, Ab, A^2b, A^3b, \ldots\}$ called a *Krylov sequence*. Then the *Krylov subspace* $\mathcal{K}_m$ is defined as
$$\mathcal{K}_m = \text{span}\{b, Ab, A^2b, \ldots, A^{m-1}b\}.$$
Arnoldi iteration can be interpreted as projection onto Krylov subspaces.

3. If A is Hermitian then everything simplifies (Hessenberg becomes tridiagonal) and we get so-called *Lanczos iteration*.

If we apply Arnoldi iteration to the solution of the linear system $Ax = b$ then we get the GMRES (generalized minimum residual) method which was proposed by Saad and Schultz in 1986.

The main idea of GMRES is to approximate the exact solution $\hat{x} = A^{-1}b$ in the m-th step by $x^{(m)} \in \mathcal{K}_m$ such that the norm of the residual

$$\|r\|_2 = \|Ax^{(m)} - b\|_2$$

is minimized. This is, of course, a least squares problem.

We want to solve the problem by using the Krylov space $\mathcal{K}_m$. If we define the $n \times m$ matrix
$$K_m = \begin{bmatrix} b & Ab & A^2b & \ldots & A^{m-1}b \end{bmatrix},$$
then a vector $x^{(m)} \in \mathcal{K}_m$ is given by

$$x^{(m)} = K_m c,$$

where c is an appropriate m-vector. Therefore

$$\|r\|_2 = \|Ax^{(m)} - b\|_2 = \|AK_m c - b\|_2.$$

We could minimize this norm by finding the QR factorization of AK_m, but that would be too expensive and also unstable. Instead, we use an orthonormal basis for $\mathcal{K}_m$, namely

$$\{q_1, q_2, \ldots, q_m\}.$$

Therefore, $x^{(m)} \in \mathcal{K}_m$ is given by

$$x^{(m)} = Q_m y, \tag{32}$$

where y is an appropriate m-vector, and minimization of the 2-norm of the residual is equivalent to minimization of

$$\|AQ_m y - b\|_2. \tag{33}$$

Our main equation for Arnoldi iteration was

$$AQ_m = Q_{m+1}\widetilde{H}_m.$$

This means that minimization of (33) is equivalent to minimization of

$$\|Q_{m+1}\widetilde{H}_m y - b\|_2. \tag{34}$$

Note that (33) involves an $n \times m$ matrix, whereas (34) only uses an $(m+1) \times m$ matrix. This means that minimization of (34) can be done more efficiently.

Since multiplication by a unitary matrix leaves the 2-norm invariant, we can further rewrite the problem, i.e.,

$$\begin{aligned} &\|Q_{m+1}\widetilde{H}_m y - b\|_2 \to \min \\ \iff &\|Q_{m+1}^* Q_{m+1}\widetilde{H}_m y - Q_{m+1}^* b\|_2 \to \min \\ \iff &\|\widetilde{H}_m y - Q_{m+1}^* b\|_2 \to \min. \end{aligned}$$

Finally, the Krylov subspaces are generated by b, i.e.,

$$\mathcal{K}_1 = \operatorname{span}\{b\}, \qquad \mathcal{K}_2 = \operatorname{span}\{b, Ab\}, \qquad \text{etc.}$$

Moreover, the columns of Q_m form an orthonormal basis for $\mathcal{K}_m$. Therefore, $q_1 = b/\|b\|$, and $q_j^* b = 0$ for all $j > 1$. Now, the entries of $Q_{m+1}^* b$ are of the form $q_j^* b$, and therefore

$$Q_{m+1}^* b = e_1 \|b\|.$$

The final version of the least squares problem to be solved for the GMRES method is thus

$$\|\widetilde{H}_m y - \|b\| e_1\|_2 \to \min, \tag{35}$$

where y is related to the solution of the linear system by (see (32))

$$x^{(m)} = Q_m y.$$

The least squares problem (35) can now be solved efficiently by QR factorization. The GMRES algorithm is

Algorithm (GMRES)

```
Input $A$, $b$
$q_1 = b/\|b\|$
for $m = 1, 2, \ldots$ do
    Perform step $m$ of Arnoldi iteration, i.e., compute the new entries for $\widetilde{H}_m$
    and $Q_m$
    Find $y$ as the minimizer of $\|\widetilde{H}_m y - \|b\| e_1\|_2$
    $x^{(m)} = Q_m y$
end
Output $x^{(m)}$
```

If the matrix-vector multiplication inside the Arnoldi algorithm can be implemented in $\mathcal{O}(n)$ operations, then the GMRES algorithm is also of $\mathcal{O}(n)$ complexity. The least squares problem is of size $(m+1) \times m$ and can be solved in $\mathcal{O}(m)$ flops provided an updating QR algorithm is used.

Similar to the conjugate gradient algorithm one can guarantee that the GMRES algorithm always converges (in exact arithmetic) in at most n steps. In this case this is guaranteed since the Krylov subspace $\mathcal{K}_n = \mathbb{C}^n = \text{range}(A)$. Moreover, since $\|r_{m+1}\| \leq \|r_m\|$ convergence is monotonic. This is due to the fact that $\|r_m\|$ is minimized over $\mathcal{K}_m$, whereas $\|r_{m+1}\|$ is minimized over the larger space $\mathcal{K}_{m+1}$.

In practice we are only interested in convergence in m steps with $m \ll n$, and then stop when the relative residual is small enough.

Finally, the rate of convergence of GMRES depends on the distribution of the eigenvalues of A (not so much on the actual condition number). The eigenvalues need to be clustered away from zero. This is illustrated by the Matlab script `GMRESDemo.m`.