Jim Lambers
CME 335
Spring Quarter 2010-11
Lecture 4 Notes

Matrices, Moments and Quadrature, cont'd

Estimation of the Regularization Parameter

Consider the least squares problem of finding $\mathbf{x}$ such that

$$\|\mathbf{c} - A\mathbf{x}\|_2^2 + \mu\|\mathbf{x}\|_2^2$$

is minimized, where μ is a parameter. This problem arises from *Tikhonov regularization* of the least-squares problem of minimizing $\|\mathbf{c} - A\mathbf{x}\|_2$.

One approach to selecting the regularization parameter μ arises from the "L-curve" that consists of the points $(\|\mathbf{x}_\mu\|_2, \|\mathbf{c} - A\mathbf{x}_\mu\|_2)$, where $\mathbf{x}_\mu$ is the solution to the above minimization problem as a function of μ. A heuristic is to find the value of μ corresponding to the "corner" of the L-curve, which obtains its name from the fact that its graph tends to be shaped like an L.

Unfortunately, computing points on the L-curve requires solving the minimization problem for several values of μ, which can be expensive. An alternative is to approximate points on the curve by expressing the coordinates of each point in terms of quadratic forms, which can then be approximated using Gaussian quadrature.

To that end, we let $K = A^T A$, and $\mathbf{d} = A^T\mathbf{d}$. Then we have

$$\|\mathbf{x}_\mu\|_2^2 = \|(A^T A + \mu I)^{-1} A^T\mathbf{c}\|_2^2 = \mathbf{d}^T(K + \mu I)^{-2}\mathbf{d},$$

$$\|\mathbf{c} - A\mathbf{x}\|_2^2 = \|\mathbf{c} - A(K + \mu I)^{-1}\mathbf{d}\|_2^2 = \mathbf{c}^T\mathbf{c} + \mathbf{d}^T K(K + \mu I)^{-2}\mathbf{d} - 2\mathbf{d}^T(K + \mu I)^{-1}\mathbf{d}.$$

The quadratic forms in these expressions that involve functions of K can be approximated by Gaussian quadrature rules obtained through Lanczos iteration applied to K with intial vector $\mathbf{d}$. Lanczos bidiagonalization, also known as Golub-Kahan bidiagonalization. can be used to obtain a Jacobi matrix from $K = A^T A$, instead of computing K and applying Lanczos iteration to it, which causes a loss of information. Further details can be found in the 1997 paper of Calvetti, Golub, and Reichel.

Estimation of the Trace of the Inverse and Determinant

Let A be an $N \times N$ symmetric positive definite matrix, with real positive eigenvalues

$$b = \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N = a.$$

Gaussian quadrature can also be used to obtain upper and lower bounds on the trace of A^{-1}, as well as $\det A$. First, we define the moments

$$\mu_r = \sum_{j=1}^{N} \lambda_i^r = \int_a^b \lambda^r \, d\alpha(\lambda).$$

Unlike in previous discussion, the measure $\alpha(\lambda)$ is now unknown. However, we do know that

$$\mu_0 = N, \quad \mu_1 = \operatorname{tr}(A), \quad \mu_2 = \operatorname{tr}(A^2) = \|A\|_F^2.$$

Our goal is to approximate $\operatorname{tr}(A^{-1}) = \mu^{-1}$ using two-node Gauss-Radau rules. Let $\tilde{\mu}_r$ be an approximation of μ_r. Then we have

$$\tilde{\mu}_r = w_0 t_0^r + w_1 t_1^r,$$

where t_0 and t_1 are Gauss-Radau nodes, and w_0 and w_1 are corresponding weights, for the measure $\alpha(\lambda)$. We assume that t_0 is the prescribed node, that is equal to either a or b, or an approximation thereof. Such a rule should have degree of accuracy 2.

The above quadrature rule has the form of a solution to a 3-term recurrence relation, where t_0 and t_1 are the roots of its characteristic equation. Because μ_0, μ_1 and μ_2 satisfy this relation, we have

$$\mu_2 + b_1 \mu_1 + b_0 \mu_0 = 0,$$

where b_0 and b_1 are the coefficients of the characteristic equation

$$z^2 + b_1 z + b_0 = (z - t_0)(z - t_1) = 0.$$

From the relations

$$b_1 = -\frac{\mu_2 + b_0 \mu_0}{\mu_1}, \quad t_1 = -t_0 - b_1, \quad b_0 = t_0 t_1,$$

we can obtain the second node t_1, and the weights w_0 and w_1 can be obtained from the relations

$$w_0 + w_1 = \mu_0, \quad w_0 t_0 + w_1 t_1 = \mu_1.$$

Finally, we obtain the approximation

$$\operatorname{tr}(A^{-1}) = \mu_{-1} \approx w_0 t_0^{-1} + w_1 t_1^{-1}.$$

If $t_0 = a$, then this approximation is an upper bound; otherwise, it is a lower bound.

From the relation

$$\ln(\det A) = \ln(\lambda_1 \lambda_2 \cdots \lambda_N) = \sum_{j=1}^{n} \ln(\lambda_j),$$

we can use the above quadrature rules with the integrand $f(\lambda) = \ln \lambda$ to obtain upper and lower bounds for $\det A$. Details can be found in the 1997 paper of Bai and Golub.

The Unsymmetric Eigenvalue Problem

We now consider the problem of computing eigenvalues of an $n \times n$ matrix A.

Power Iterations

For simplicity, we assume that A has eigenvalues $\lambda_1, \ldots, \lambda_n$ such that

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|.$$

We also assume that A is diagonalizable, meaning that it has n linearly independent eigenvectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ such that $A\mathbf{x}_i = \lambda_i \mathbf{x}_i$ for $i = 1, \ldots, n$.

Suppose that we continually multiply a given vector $\mathbf{x}^{(0)}$ by A, generating a sequence of vectors $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots$ defined by

$$\mathbf{x}^{(k)} = A\mathbf{x}^{(k-1)} = A^k \mathbf{x}^{(0)}, \quad k = 1, 2, \ldots.$$

Because A is diagonalizable, any vector in $\mathbb{R}^n$ is a linear combination of the eigenvectors, and therefore we can write $\mathbf{x}^{(0)} = c_1\mathbf{x}_1 + c_2\mathbf{x}_2 + \cdots + c_n\mathbf{x}_n$. We then have

$$\begin{aligned}
\mathbf{x}^{(k)} &= A^k \mathbf{x}^{(0)} \\
&= \sum_{i=1}^{n} c_i A^k \mathbf{x}_i \\
&= \sum_{i=1}^{n} c_i \lambda_i^k \mathbf{x}_i \\
&= \lambda_1^k \left[c_1 \mathbf{x}_1 + \sum_{i=2}^{n} c_i \left(\frac{\lambda_i}{\lambda_1} \right)^k \mathbf{x}_i \right].
\end{aligned}$$

Because $|\lambda_1| > |\lambda_i|$ for $i = 2, \ldots, n$, it follows that the coefficients of $\mathbf{x}_i$, for $i = 2, \ldots, n$, converge to zero as $k \to \infty$. Therefore, the direction of $\mathbf{x}^{(k)}$ converges to that of $\mathbf{x}_1$. This leads to the most basic method of computing an eigenvalue and eigenvector, the *Power Method:*

Choose an initial vector $\mathbf{q}_0$ such that $\|\mathbf{q}_0\|_2 = 1$
for $k = 1, 2, \ldots$ **do**
    $\mathbf{z}_k = A\mathbf{q}_{k-1}$
    $\mathbf{q}_k = \mathbf{z}_k / \|\mathbf{z}_k\|_2$
end

This algorithm continues until $\mathbf{q}_k$ converges to within some tolerance. If it converges, it converges to a unit vector that is a scalar multiple of $\mathbf{x}_1$, an eigenvector corresponding to the largest eigenvalue, λ_1. The rate of convergence is $|\lambda_1/\lambda_2|$, meaning that the distance between $\mathbf{q}_k$ and a vector parallel to $\mathbf{x}_1$ decreases by roughly this factor from iteration to iteration.

It follows that convergence can be slow if λ_2 is almost as large as λ_1, and in fact, the power method fails to converge if $|\lambda_2| = |\lambda_1|$, but $\lambda_2 \neq \lambda_1$ (for example, if they have opposite signs). It is worth noting the implementation detail that if λ_1 is negative, for example, it may appear that $\mathbf{q}_k$ is not converging, as it "flip-flops" between two vectors. This is remedied by normalizing $\mathbf{q}_k$ so that it is not only a unit vector, but also a positive number.

Once the normalized eigenvector $\mathbf{x}_1$ is found, the corresponding eigenvalue λ_1 can be computed using a Rayleigh quotient. Then, *deflation* can be carried out by constructing a Householder reflection P_1 so that $P_1\mathbf{x}_1 = \mathbf{e}_1$, as discussed previously, and then P_1AP_1 is a matrix with block upper-triangular structure. This decouples the problem of computing the eigenvalues of A into the (solved) problem of computing λ_1, and then computing the remaining eigenvalues by focusing on the lower right $(n-1) \times (n-1)$ submatrix.

This method can be impractical, however, due to the contamination of smaller eigenvalues by roundoff error from computing the larger ones and then deflating. An alternative is to compute several eigenvalues "at once" by using a block version of the Power Method, called *Orthogonal Iteration*. In this method, A is multiplied by an $n \times r$ matrix, with $r > 1$, and then the normalization of the vector computed by the power method is generalized to the *orthogonalization* of the block, through the QR factorization. The method is as follows:

Choose an $n \times r$ matrix Q_0 such that $Q_0^H Q_0 = I_r$
for $k = 1, 2, \ldots$ **do**
  $Z_k = AQ_{k-1}$
  $Z_k = Q_k R_k$ (QR Factorization)
end

Generally, this method computes a convergent sequence $\{Q_k\}$, as long as Q_0 is not deficient in the directions of certain eigenvectors of A^H, and $|\lambda_r| > |\lambda_{r+1}|$. From the relationship

$$R_k = Q_k^H Z_k = Q_k^H A Q_{k-1},$$

we see that if Q_k converges to a matrix Q, then $Q^H AQ = R$ is upper-triangular, and because $AQ = QR$, the columns of Q span an invariant subspace.

Furthermore, if $Q^\perp$ is a matrix whose columns span $(\text{range}(Q))^\perp$, then

$$\begin{bmatrix} Q^H \\ (Q^\perp)^H \end{bmatrix} A \begin{bmatrix} Q & Q^\perp \end{bmatrix} = \begin{bmatrix} Q^H AQ & Q^H AQ^\perp \\ (Q^\perp)^H AQ & (Q^\perp)^H AQ^\perp \end{bmatrix} = \begin{bmatrix} R & Q^H AQ^\perp \\ 0 & (Q^\perp)^H AQ^\perp \end{bmatrix}.$$

That is, $\lambda(A) = \lambda(R) \cup \lambda((Q^\perp)^H AQ^\perp)$, and because R is upper-triangular, the eigenvalues of R are its diagonal elements. We conclude that Orthogonal Iteration, when it converges, yields the largest r eigenvalues of A.

If we let $r = n$, then, if the eigenvalues of A are separated in magnitude, then generally Orthogonal Iteration converges, yielding the Schur Decomposition of A, $A = QTQ^H$. However,

this convergence is generally quite slow. Before determining how convergence can be accelerated, we examine this instance of Orthogonal Iteration more closely.

For each integer k, we define $T_k = Q_k^H A Q_k$. Then, from the algorithm for Orthogonal Iteration, we have

$$T_{k-1} = Q_{k-1}^H A Q_{k-1} = Q_{k-1}^H Z_k = (Q_{k-1}^H Q_k) R_k,$$

and

$$\begin{aligned} T_k &= Q_k^H A Q_k \\ &= Q_k^H A Q_{k-1} Q_{k-1}^H Q_k \\ &= Q_k^H Z_k Q_{k-1}^H Q_k \\ &= Q_k^H Q_k R_k (Q_{k-1}^H Q_k) \\ &= R_k (Q_{k-1}^H Q_k). \end{aligned}$$

That is, T_k is obtained from T_{k-1} by computing the QR factorization of T_{k-1}, and then multiplying the factors in reverse order. Equivalently, T_k is obtained by applying a unitary similarity transformation to T_{k-1}, as

$$T_k = R_k (Q_{k-1}^H Q_k) = (Q_{k-1}^H Q_k)^H T_{k-1} (Q_{k-1}^H Q_k) = U_k^H T_{k-1} U_k.$$

If Orthogonal Iteration converges, then T_k converges to an upper-triangular matrix $T = Q^H A Q$ whose diagonal elements are the eigenvalues of A. This simple process of repeatedly computing the QR factorization and multiplying the factors in reverse order is called the *QR Iteration*, which proceeds as follows:

Choose Q_0 so that $Q_0^H Q_0 = I_n$ $T_0 = Q_0^H A Q_0$
for $k = 1, 2, \ldots$ **do**
    $T_{k-1} = U_k R_k$ (QR factorization)
    $T_k = R_k U_k$
end

It is this version of Orthogonal Iteration that serves as the cornerstone of an efficient algorithm for computing all of the eigenvalues of a matrix. As described, QR iteration is prohibitively expensive, because $O(n^3)$ operations are required in *each* iteration to compute the QR factorization of T_{k-1}, and typically, many iterations are needed to obtain convergence. However, we will see that with a judicious choice of Q_0, the amount of computational effort can be drastically reduced.

It should be noted that if A is a real matrix with complex eigenvalues, then Orthogonal Iteration or the QR Iteration will not converge, due to distinct eigenvalues having equal magnitude. However, the *structure* of the matrix T_k in QR Iteration generally will converge to "quasi-upper-triangular" form, with 1×1 or 2×2 diagonal blocks corresponding to real eigenvalues or complex-conjugate pairs of eigenvalues, respectively. It is this type of convergence that we will seek in our continued development of the QR Iteration.

The Hessenberg and Real Schur Forms

Let A be a real $n \times n$ matrix. It is possible that A has complex eigenvalues, which must occur in complex-conjugate pairs, meaning that if $a + ib$ is an eigenvalue, where a and b are real, then so is $a - ib$. On the one hand, it is preferable that complex arithmetic be avoided as much as possible when using QR iteration to obtain the Schur Decomposition of A. On the other hand, in the algorithm for QR iteration, if the matrix Q_0 used to compute $T_0 = Q_0^H A Q_0$ is real, then every matrix T_k generated by the iteration will also be real, so it will not be possible to obtain the Schur Decomposition.

We compromise by instead seeking to compute the *Real Schur Decomposition* $A = QTQ^T$ where Q is a real, orthogonal matrix and T is a real, *quasi-upper-triangular* matrix that has a *block* upper-triangular structure

$$T = \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1p} \\ 0 & T_{22} & \ddots & \vdots \\ 0 & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & T_{pp} \end{bmatrix},$$

where each diagonal block T_{ii} is 1×1, corresponding to a real eigenvalue, or a 2×2 block, corresponding to a pair of complex eigenvalues that are conjugates of one another.

If QR iteration is applied to such a matrix, then the sequence $\{T_k\}$ will not converge, but a block upper-triangular structure will be obtained, which can then be used to compute all of the eigenvalues. Therefore, the iteration can be terminated when appropriate entries below the diagonal have been made sufficiently small.

However, one significant drawback to the QR iteration is that each iteration is too expensive, as it requires $O(n^3)$ operations to compute the QR factorization, and to multiply the factors in reverse order. Therefore, it is desirable to first use a similarity transformation $H = U^T A U$ to reduce A to a form for which the QR factorization and matrix multiplication can be performed more efficiently.

Suppose that U^T includes a Householder reflection, or a product of Givens rotations, that transforms the first column of A to a multiple of $\mathbf{e}_1$, as in algorithms to compute the QR factorization. Then U operates on all rows of A, so when U is applied to the columns of A, to complete the similarity transformation, it affects all columns. Therefore, the work of zeroing the elements of the first column of A is undone.

Now, suppose that instead, U^T is designed to zero all elements of the first column except the first *two*. Then, U^T affects all rows except the first, meaning that when $U^T A$ is multiplied by U on the right, the first column is *unaffected*. Continuing this reasoning with subsequent columns of A, we see that a sequence of orthogonal transformations can be used to reduce A to an *upper Hessenberg* matrix H, in which $h_{ij} = 0$ whenever $i > j+1$. That is, all entries below the *subdiagonal* are equal to zero.

It is particularly efficient to compute the QR factorization of an upper Hessenberg, or simply Hessenberg, matrix, because it is only necessary to zero one element in each column. Therefore,

it can be accomplished with a sequence of $n-1$ Givens row rotations, which requires only $O(n^2)$ operations. Then, these same Givens rotations can be applied, in the same order, to the columns in order to complete the similarity transformation, or, equivalently, accomplish the task of multiplying the factors of the QR factorization.

Specifically, given a Hessenberg matrix H, we apply Givens row rotations $G_1^T, G_2^T, \ldots, G_{n-1}^T$ to H, where G_i^T rotates rows i and $i+1$, to obtain

$$G_{n-1}^T \cdots G_2^T G_1^T H = (G_1 G_2 \cdots G_{n-1})^T H = Q^T H = R,$$

where R is upper-triangular. Then, we compute

$$\tilde{H} = Q^T H Q = RQ = RG_1 G_2 \cdots G_{n-1}$$

by applying column rotations to R, to obtain a new matrix $\tilde{H}$.

By considering which rows or columns the Givens rotations affect, it can be shown that Q is Hessenberg, and therefore $\tilde{H}$ is Hessenberg as well. The process of applying an orthogonal similarity transformation to a Hessenberg matrix to obtain a new Hessenberg matrix with the same eigenvalues that, hopefully, is closer to quasi-upper-triangular form is called a *Hessenberg QR step*. The following algorithm overwrites H with $\tilde{H} = RQ = Q^T H Q$, and also computes Q as a product of Givens column rotations, which is only necessary if the full Schur Decomposition of A is required, as opposed to only the eigenvalues.

for $j = 1, 2, \ldots, n-1$ **do**
    $[c, s] = \texttt{givens}(h_{jj}, h_{j+1,j})$
    $G_j = \begin{bmatrix} c & -s \\ s & c \end{bmatrix}$
    $H(j:j+1, j:n) = G_j^T H(j:j+1, :)$
end
$Q = I$
for $j = 1, 2, \ldots, n-1$ **do**
    $H(1:j+1, j:j+1) = H(1:j+1, j:j+1) G_j$
    $Q(1:j+1, j+j+1) = Q(1:j+1, j:j+1) G_j$
end

The function $\texttt{givens}(a, b)$ returns c and s such that

$$\begin{bmatrix} c & -s \\ s & c \end{bmatrix}^T \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}, \quad r = \sqrt{a^2 + b^2}.$$

Note that when performing row rotations, it is only necessary to update certain columns, and when performing column rotations, it is only necessary to update certain rows, because of the

structure of the matrix at the time the rotation is performed; for example, after the first loop, H is upper-triangular.

Before a Hessenberg QR step can be performed, it is necessary to actually reduce the original matrix A to Hessenberg form $H = U^T AU$. This can be accomplished by performing a sequence of Householder reflections $U = P_1 P_2 \cdots P_{n-2}$ on the columns of A, as in the following algorithm.

$U = I$
for $j = 1, 2, \ldots, n-2$ **do**
  $\mathbf{v} = \texttt{house}(A(j+1:n, j))$, $c = 2/\mathbf{v}^T\mathbf{v}$
  $A(j+1:n, j:n) = A(j+1:n, j:n) - c\mathbf{v}\mathbf{v}^T A(j+1:n, j:n)$
  $A(1:n, j+1:n) = A(1:n, j+1:n) - cA(1:n, j+1:n)\mathbf{v}\mathbf{v}^T$
end

The function $\texttt{house}(\mathbf{x})$ computes a vector $\mathbf{v}$ such that $P\mathbf{x} = I - c\mathbf{v}\mathbf{v}^T\mathbf{x} = \alpha\mathbf{e}_1$, where $c = 2/\mathbf{v}^T\mathbf{v}$ and $\alpha = \pm\|\mathbf{x}\|_2$. The algorithm for the Hessenberg reduction requires $O(n^3)$ operations, but it is performed only once, before the QR Iteration begins, so it still leads to a substantial reduction in the total number of operations that must be performed to compute the Schur Decomposition.

If a subdiagonal entry $h_{j+1,j}$ of a Hessenberg matrix H is equal to zero, then the problem of computing the eigenvalues of H decouples into two smaller problems of computing the eigenvalues of H_{11} and H_{22}, where

$$H = \begin{bmatrix} H_{11} & H_{12} \\ 0 & H_{22} \end{bmatrix}$$

and H_{11} is $j \times j$. Therefore, an efficient implementation of the QR Iteration on a Hessenberg matrix H focuses on a submatrix of H that is *unreduced*, meaning that all of its subdiagonal entries are nonzero. It is also important to monitor the subdiagonal entries after each iteration, to determine if any of them have become nearly zero, thus allowing further decoupling. Once no further decoupling is possible, H has been reduced to quasi-upper-triangular form and the QR Iteration can terminate.

It is essential to choose an *maximal* unreduced diagonal block of H for applying a Hessenberg QR step. That is, the step must be applied to a submatrix H_{22} such that H has the structure

$$H = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ 0 & H_{22} & H_{23} \\ 0 & 0 & H_{33} \end{bmatrix}$$

where H_{22} is unreduced. This condition ensures that the eigenvalues of H_{22} are also eigenvalues of H, as $\lambda(H) = \lambda(H_{11}) \cup \lambda(H_{22}) \cup \lambda(H_{33})$ when H is structured as above. Note that the size of either H_{11} or H_{33} may be 0×0.

The following property of unreduced Hessenberg matrices is useful for improving the efficiency of a Hessenberg QR step.

Theorem (Implicit Q Theorem) Let A be an $n \times n$ matrix, and let Q and P be $n \times n$ orthgonal matrices such that $Q^T AQ = H$ and $V^T AV = G$ are both upper Hessenberg, and H is unreduced.

If $Q = \begin{bmatrix} \mathbf{q}_1 & \cdots & \mathbf{q}_n \end{bmatrix}$ and $V = \begin{bmatrix} \mathbf{v}_1 & \cdots & \mathbf{v}_n \end{bmatrix}$, and if $\mathbf{q}_1 = \mathbf{v}_1$, then $\mathbf{q}_i = \pm\mathbf{v}_i$ for $i = 2, \ldots, n$, and $|h_{ij}| = |g_{ij}|$ for $i, j = 1, 2, \ldots, n$.

That is, if two orthogonal similarity transformations that reduce A to Hessenberg form have the same first column, then they are "essentially equal", as are the Hessenberg matrices.

The proof of the Implicit Q Theorem proceeds as follows: From the relations $Q^TAQ = H$ and $V^TAV = G$, we obtain $GW = WH$, where $W = V^TQ$ is orthogonal. Because $\mathbf{q}_1 = \mathbf{v}_1$, we have $W\mathbf{e}_1 = \mathbf{e}_1$. Equating first columns of $GW = WH$, and keeping in mind that G and H are both upper Hessenberg, we find that only the first two elements of $W\mathbf{e}_2$ are nonzero. Proceeding by induction, it follows that W is upper triangular, and therefore W^{-1} is also upper triangular. However, because W is orthogonal, $W^{-1} = W^T$, which means that W^{-1} is lower triangular as well. Therefore, W is a diagonal matrix, so by the orthogonality of W, W must have diagonal entries that are equal to ± 1, and the theorem follows.

Another important property of an unreduced Hessenberg matrix is that all of its eigenvalues have a geometric multiplicity of one. To see this, consider the matrix $H - \lambda I$, where H is an $n \times n$ unreduced Hessenberg matrix and λ is an arbitrary scalar. If λ is not an eigenvalue of H, then H is nonsingular and $\text{rank}(H) = n$. Otherwise, because H is unreduced, from the structure of H it can be seen that the first $n - 1$ columns of $H - \lambda I$ must be linearly independent. We conclude that $\text{rank}(H - \lambda I) = n - 1$, and therefore at most one vector $\mathbf{x}$ (up to a scalar multiple) satisfies the equation $H\mathbf{x} = \lambda\mathbf{x}$. That is, there can only be one linearly independent eigenvector. It follows that if any eigenvalue of H repeats, then it is defective.