Department of Mathematics, University of Utah

Analysis of Numerical Methods I

MTH6610 – Section 001 – Fall 2017

Lecture notes – Rayleigh iteration

Monday, October 30, 2017

These notes are not a substitute for class attendance. Their main purpose is to provide a lecture overview summarizing the topics covered.

Reading: Trefethen & Bau III, Lectures 27

The contents of this set of lecture notes are a high-level description of a useful algorithm for computing eigenvalues: Rayleigh iteration. This algorithm essentially uses two ideas in tandem, so we first describe these ideas.

Throughout, we assume A is a square, $n \times n$ real-valued symmetric matrix (so $A^* = A^T$) with a complete set of eigenvectors. (Hence it is not a defective matrix.) We let $\lambda_1, \ldots, \lambda_n$ denote its (real-valued) eigenvalues associated to its (real-valued) eigenvectors $v_1, \ldots, v_n$.

One tool is very useful for approximating an eigenvalue if an approximation to its corresponding eigenvector is known: the Rayleigh quotient. The Rayleigh quotient is defined, for $x \in \mathbb{R}^n$, as

$$r(x) = \frac{x^T A x}{x^T x}$$

This is a real-valued function. If x equals, say v_j, then $r(x) = \lambda_j$. A computation also shows that

$$\frac{\partial r}{\partial x_j} = \frac{2}{\|x\|^2}\left((Ax)_j - r(x) x_j\right),$$

where x_j is the jth component of the vector x. This shows in particular that $\nabla r(v_j) = 0$, i.e., that the gradient of r vanishes at an eigenvector of A. Because of this, a Taylor's Theorem argument shows that, if $x \approx v_j$, then

$$r(x) - \lambda_j = \mathcal{O}\left(\|x - v_j\|^2\right).$$

This is utility of Rayleigh quotients: they provide an estimate of an eigenvalue is that quadratic compared to the error in the approximate eigenvector.

Rayleigh quotients can be used to approximate eigenvalues given approximate eigenvectors. The second ingredient we need is the ability to approximate eigenvectors given approximate eigenvalues.

The first algorithm algorithm we describe is a very well-known one: power iteration. Suppose that, in magnitude, λ_1 is larger than all the other eigenvalues, i.e.,

$$|\lambda_1| \geq |\lambda_j|, \qquad\qquad j = 2, \ldots, n.$$

Now let v be an arbitrary vector in $\mathbb{R}^n$. Since A has a full set of eigenvectors, then there exist constants $c_1, \ldots, c_n$ such that

$$v = \sum_{j=1}^{n} c_j v_j.$$

Then we have

$$Av = \sum_{j=1}^{n} c_j A v_j = \sum_{j=1}^{n} c_j \lambda_j v_j.$$

Applying A a total of k times yields

$$A^k v = \sum_{j=1}^{n} c_j \lambda_j^k v_j = \lambda_1^k \left(c_1 v_1 + \sum_{j=2}^{n} c_j \left(\frac{\lambda_j}{\lambda_1}\right)^k v_j \right).$$

Since $|\lambda_j/\lambda_1|^k \to 0$ for large k, then $A^k v$ is approximately v_1. One can see from the computation that we expect

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

While this seems like a useful result, the fact that we converge only like $|\lambda_2/\lambda_1|$ makes this algorithm limited, especially in cases when these two eigenvalues are similar in magnitude. The method if *inverse iteration* seeks to fix the problem in power iteration when eigenvalues are clustered together in magnitude. The essential difference is that inverse iteration uses a transformation to amplify ratios of eigenvalues. Assume μ is not an eigenvalue of A. If v_j is an eigenvector of A associated to λ_j, then v_j is also an eigenvector of $(A-\mu I)^{-1}$ associated to eigenvalue $(\lambda_j - \mu)^{-1}$. However, if μ is very close to λ_j, then

$$\left| \frac{1}{\lambda_j - \mu} \right| \gg \left| \frac{1}{\lambda_k - \mu} \right|, \qquad\qquad k \neq j.$$

I.e., the matrix $B = (A - \mu I)^{-1}$ has a dominant eigenvalue that is very far away from the remaining eigenvalues. Therefore, power iteration on this matrix should be effective. The method of inverse iteration simply uses power iteration on the matrix B. For enough iterations, this provides a good estimate to an eigenvector of A if μ is close to an eigenvalue of A.

The main algorithm for this lecture, Rayleigh iteration, is a combination of the Rayleigh quotient method for approximating eigenvalues, with the inverse iteration method for approximating eigenvectors. Rayleigh iteration is simply inverse iteration where one updates the value of μ using a Rayleigh quotient. Let $v \in \mathbb{R}^n$ and $\mu \in \mathbb{R}$ be arbitrary. Then repeated perform the updates

$$w \coloneqq (A - \mu I)^{-1} v, \qquad\qquad v = w/\|w\|, \qquad\qquad \mu = v^T A v.$$

The first operation is inverse iteration, where w approximates an eigenvector associated with the eigenvalue close to μ. The third operation updates μ via a Rayleigh quotient estimate from w. The process simply repeats. This is Rayleigh iteration, and its convergence is cubic: if $v^{(k)}$ and $\mu^{(k)}$ are the estimates at the kth iteration, then

$$\left\| v^{(k+1)} - v_j \right\| \lesssim \left\| v^{(k)} - v_j \right\|^3, \qquad\qquad \left| \mu^{(k+1)} - \lambda_j \right| \lesssim \left| \mu^{(k)} - \lambda_j \right|^3.$$

where j is the index associated to the eigenvalue closest to the starting vector.