Spectral Graph Theory and its Applications Lecture 18

Solving Linear Equations

Lecturer: Daniel A. Spielman November 9, 2004

18.1 Introduction

This lecture is about solving linear equations of the form $Ax = b$, where A is a symmetric positive semi-definite matrix. In fact, we will usually assume that A is diagonally dominant, and will sometimes assume that it is in fact positive definite.

We will briefly describe the Chebyshev method for solving such systems, and the related Conjugate Gradient. We will then explain how these methods can be accelerated by preconditioning. In the next few lectures, we will see how material from the class can be applied to construct good preconditioners.

18.2 Approximate Inverses

In undergraduate linear algebra classes, we are taught to solve a system like $Ax = b$ by first computing the inverse of A, call it C, and then setting $x = Cb$. However, this can take way too long. In this lecture, we will see some ways of quickly computing approximate inverses of A, when A is symmetric and positive semi-definite. By "quickly", we mean that we might not actually compute the entries of the approximate inverse, but rather just provide a fast procedure for multiplying it by a vector. I'll talk more about that part later, so let's first consider a reasonable notion of approximate.

The fact that the inverse C satisfies $AC = I$ is equivalent to saying that all eigenvalues of AC are 1. We will call C an ϵ-approximate inverse of A if all eigenvalues of AC lie in $[1 - \epsilon, 1 + \epsilon]$. In this case, all eigenvalues of $I - AC$ lie in $[-\epsilon, \epsilon]$, from which we can see that C can be used to approximately solve linear systems in A. If we set $x = Cb$, then

$$\|b - Ax\| = \|b(I - AC)\| .$$

In our construction, we will guarantee that AC is symmetric, and so $\|I - AC\| \leq \epsilon$ and

$$\|b(I - AC)\| \leq \|b\| \|I - AC\| \leq \epsilon \|b\| .$$

18.3 Chebyshev Polynomials

We will exploit matrices C of the form

$$C = c_0 I + c_1 A + c_2 A^2 + \cdots + c_t A^t .$$

Note that one can quickly compute the product Cb: we will iteratiely compute $A^i b$, and then take the sum. So, if A has m non-zero entries, then this will take $O(mt)$ steps. I should warn you that I'm ignoring some numerical issues here.

Now, let's find a good choice of polynomials. Recall that we want

$$I - AC = I - c_0 A - c_1 A^2 - \cdots - c_t A^{t+1}$$

to have all its eigenvalues between $-\epsilon$ and ϵ. Let

$$p(x) \stackrel{\text{def}}{=} 1 - c_0 x - c_1 x^2 - \cdots - c^t x^{t+1}.$$

Then, for each eigenvalue λ of A, $p(\lambda)$ is an eigenvalue of $I - AC$. Assuming that A is positive definite, its eigenvalues all lie between λ_{min} and λ_{max}, where $\lambda_{min} > 0$. So, it suffices to find a polynomial p with constant term 1 such that $|p(x)| < \epsilon$ for all $x \in [\lambda_{min}, \lambda_{max}]$. As in Lecture 8, we will use Chebyshev polynomials to construct p. We recall that the k-th Chebyshev polynomial, T_k, has the following properties

1. T_k has degree k.
2. $T_k(x) \in [-1, 1]$, for $x \in [-1, 1]$.
3. $T_k(x)$ is monotonically increasing for $x \geq 1$.
4. $T_k(1 + \gamma) \geq (1 + \sqrt{2\gamma})^k / 2$, for $\gamma > 0$.

To express $p(x)$ in terms of a Chebyshev polynomial, we should map the range on which we want p to be small, $[\lambda_{min}, \lambda_{max}]$ to $[-1, 1]$. We will accomplish this with the linear map:

$$l(x) \stackrel{\text{def}}{=} \frac{\lambda_{max} + \lambda_{min} - 2x}{\lambda_{max} - \lambda_{min}}.$$

Note that

$$l(x) = \begin{cases} -1 & \text{if } x = \lambda_{max} \\ 1 & \text{if } x = \lambda_{min} \\ \frac{\lambda_{max} + \lambda_{min}}{\lambda_{max} - \lambda_{min}} & \text{if } x = 0. \end{cases}$$

So, to guarantee that the constant coefficient in $p(x)$ is zero, we should set

$$p(x) \stackrel{\text{def}}{=} \frac{T_t(l(x))}{T_t(l(0))}.$$

We know that $|T_t(l(x))| \leq 1$ for $x \in [\lambda_{min}, \lambda_{max}]$. To find $p(x)$ for x in this range, we must compute $T_t(l(0))$. We will express the result of our computation in terms of the condition number of, which for a symmetric matrix is

$$\kappa(A) \stackrel{\text{def}}{=} \frac{\lambda_{max}(A)}{\lambda_{min}(A)}.$$

We have

$$l(0) \geq 1 + 2/\kappa(A),$$

and so by properties 3 and 4 of Chebyshev polynomials,

$$T_t(l(0)) \geq (1 + 2/\sqrt{\kappa})^k/2.$$

Thus,

$$p(x) \leq 2(1 + 2/\sqrt{\kappa})^k,$$

for $x \in [\lambda_{min}, \lambda_{max}]$, and so all eigenvalues of $I - AC$ will have absolute value at most $2(1+2/\sqrt{\kappa})^k$. So, to get accuracy ϵ in our solution to $Ax = b$, we need $t = O(\sqrt{1/\kappa} \log \epsilon)$.

18.4 Computational properties, and the Conjugate Gradient

The Chebyshev properties have some useful properties that I should mention, but which I will not prove. The most useful property is that their coefficients have a clean recursive definition. As a result, we can construct an algorithm, called the Chebyshev method, that just takes as input a lower bound on λ_{min} and an upper bound on λ_{max} and, after t iterations produces the estimate obtained from T_t. In particular, the algorithm is iterative, and at each iteration obtain the correct estimate.

The Chebyshev method is not used much in practice because there is a better method—the Conjugate Gradient. The CG method adaptively computes the coefficients of the polynomial, and is guaranteed to compute the optimal coefficients at each iteration. I will now dwell on how it works now, but I will mention that it is quite efficient, and always at least as good as the Chebyshev method.

18.5 Preconditioning

So, we know that we can obtain an approximate solution to a system of the form $Ax = b$ in time depending on $\sqrt{\kappa(A)}$. But, what if $\kappa(A)$ is large? In this case, we can try to *precondition* A. That means finding a matrix B that is similar to A, but for which it is easy to solve equations like $Bz = c$. We then apply the Chebyshev method, or the Conjugate Gradient, to solve

$$B^{-1}Ax = B^{-1}b.$$

The running time of the preconditioned method is effected in two ways:

1. At each iteration, in addition to multiplying a vector by A, we must solve a linear system $Bz = c$, which we can think of as multiplying by B^{-1}, and
2. The number of iterations now depends the eigenvalues of $\rho(B^{-1}A)$. If A and B are positive definite, then all the eigenvalues of $B^{-1}A$ will be real and non-negative, so the number of iterations just depends upon

$$\kappa(A, B) = \lambda_{max}(B^{-1}A)/\lambda_{min}(B^{-1}A).$$

Note that I am ignoring one detail that $B^{-1}A$ is not symmetric. Please just trust me for now that this doesn't make much difference, so I can get through the rest of the lecture.

Our goal in preconditioning is to find a matrix B that guarantees that the eigenvalues of $B^{-1}A$ are not too big, and so that it is easy to solve a linear system in B. It will help if I give a different characterization of $\kappa(A, B)$.