Course Notes for EE227C (Spring 2018): Convex Optimization and Approximation

Instructor: Moritz Hardt

Email: hardt+ee227c@berkeley.edu

Graduate Instructor: Max Simchowitz

Email: msimchow+ee227c@berkeley.edu

February 18, 2018

8 Lecture 8: Conjugate gradients and Krylov subspaces

In this lecture, we'll develop a unified view of solving linear equations $Ax = b$ and eigenvalue problems $Ax = \lambda x$. In particular, we will justify the following picture.

	$Ax = b$	$Ax = \lambda x$
Basic	Gradient descent	Power method
Accelerated	Chebyshev iteration	Chebyshev iteration
Accelerated and step size free	Conjugate gradient	Lanczos

What we saw last time was the basic gradient descent method and Chebyshev iteration for solving quadratics. Chebyshev iteration requires step sizes to be carefully chosen. In this section, we will see how we can get a "step-size free" accelerated method, known as *conjugate gradient*.

What ties this all together is the notion of a Krylov subspace and its corresponding connection to low-degree polynomials.

Our exposition follows the excellent Chapter VI in Trefethen-Bau [TD97].

8.1 Krylov subspaces

The methods we discuss all have the property that they generate a sequence of points iteratively that is contained in a subspace called the *Kyrlov subspace*.

Definition 8.1 (Krylov subspace)**.** For a matrix $A \in \mathbb{R}^{nxn}$ and a vector $b \in \mathbb{R}^n$, the *Krylov sequence* of order t is $b, Ab, A^2b, ..., A^tb$. We define the *Krylov subspace* as

$$K_t(A, b) = \text{span}(\{b, Ab, A^2b, \ldots, A^t\}) \subseteq \mathbb{R}^n \,.$$

Krylov subspace naturally connect to polynomial approximation problems. To see this, recall that a degree t matrix polynomial is an expression of the form $p(A) = \sum_{i=1}^{t} \alpha_i A^i$.

Fact 8.2 (Polynomial connection)**.** *The Krylov subspace satisfies*

$$K_t(A, b) = \{p(A)b : deg(p) \leqslant t\} \,.$$

Proof. Note that

$$v \in K_t(A, b) \iff \exists \alpha_i : \; v = \alpha_0 b + \alpha_1 Ab + \cdots \alpha_t A^t b$$

■

From here on, suppose we have a symmetric matrix $A \in \mathbb{R}^{n \times n}$ that has orthonormal eigenvectors $u_1 \ldots u_n$ and ordered eigenvalues $\lambda_1 \geqslant \lambda_2 \ldots \geqslant \lambda_n$. Recall, this means

$$\begin{aligned} \langle u_i, u_j \rangle &= 0, \quad \text{for } i \neq j \\ \langle u_i, u_i \rangle &= 1 \end{aligned}$$

Using that $A = \sum_i \lambda_i u_i u_i^\top$, it follows

$$p(A)u_i = p(\lambda_i)u_i \,.$$

Now suppose we write b in the eigenbasis of A as

$$b = \alpha_1 u_1 + ... + \alpha_n u_n$$

with $\alpha_i = \langle u_i, b \rangle$. It follows that

$$p(A)b = \alpha_1 p(\lambda_1)u_1 + \alpha_2 p(\lambda_2)u_2 + \ldots + \alpha_n p(\lambda_n)u_n \,.$$

8.2 Finding eigenvectors

Given these ideas, one natural approach to finding eigenvectors is to find a polynomial p such that

$$p(A)b \approx \alpha_1 u_1 \,.$$

Ideally, we would have $p(\lambda_1) = 1$ and $p(\lambda_i) = 0$ for $i > 1$, but this is in general impossible unless we maket he degree of our polynomial as high as the number of distinct eigenvalues of A. Keep in mind that the degree ultimately determines the

number of steps that our iterative algorithm makes. We'd therefore like to keep it as small as possible.

That's why we'll settle for an approximate solution that has $p(\lambda_1) = 1$ and makes $\max_{i>1} p(\lambda_i)$ as small as possible. This will give us a close approximation to the top eigenvalue. In practice, we don't know the value λ_1 ahead of time. What we therefore really care about is the ratio $p(\lambda_1)/p(\lambda_2)$ so that no matter what λ_1, the second eigenvalue will get mapped to a much smaller value by p.

We consider the following simple polynomial $p(\lambda) = \lambda^t$ that satisfies

$$p(\lambda_2)/p(\lambda_1) = \left(\frac{\lambda_2}{\lambda_1}\right)^t$$

In the case where $\lambda_1 = (1+\epsilon)\lambda_2$ we need $t = O(1/\epsilon)$ to make the ratio small.

The next lemma turns a small ratio into an approximation result for the top eigenvector. To state the lemma, we recall that $\tan\angle(a,b)$ is the tangent of the angle between a and b.

Lemma 8.3. $\tan\angle(p(A)b, u_1) \leqslant \max_{j>1} \frac{|p(\lambda_j)|}{|p(\lambda_1)|} \tan\angle(b, u_1)$

Proof. We define $\theta = \angle(u_1, b)$. By this, we get

$$\begin{aligned}
\sin^2\theta &= \sum_{j>1} \alpha_j^2 \\
\cos^2\theta &= |\alpha_1|^2 \\
\tan^2\theta &= \sum_{j>1} \frac{|\alpha_j^2|}{|\alpha_1|^2}
\end{aligned}$$

Now we can write:

$$\tan^2\angle(p(A)b, u_1) = \sum_{j>1} \frac{|p(\lambda_j)\alpha_j|^2}{|p(\lambda_1)\alpha_1|^2} \leqslant \max_{j>1} \frac{|p(\lambda_j)|^2}{|p(\lambda_1)|^2} \sum_{j>1} \frac{\alpha_j|^2}{|\alpha_1|^2}$$

We note that this last sum $\sum_{j>1} \frac{\alpha_j|^2}{|\alpha_1|^2} = \tan\theta$ and we obtain our desired result. ∎

Applying the lemma to $p(\lambda) = \lambda^t$ and $\lambda_1 = (1+\epsilon)\lambda_2$, we get

$$\tan\angle(p(A)b, u_1) \leqslant (1+\epsilon)^{-t} \tan\angle(u_1, b)\,.$$

If there is a big gap between λ_1 and λ_2 this converges quickly but it can be slow if $\lambda_1 \approx \lambda_2$. It worth noting that if we choose $b \in \mathbb{R}^n$ to be a random direction, then

$$\mathbb{E}\left[\tan\angle(u_1, b)\right] = O\left(\sqrt{n}\right)\,.$$

Going one step further we can also see that the expression $p(A)b = A^t b$ can of course be built iteratively by repeatedly multiplying by A. For reasons of numerical stability it

makes sense to normalize after each matrix-vector multiplication. This preserved the direction of the iterate and therefore does not change our convergence analysis. The resulting algorithms is the well known power method, defined recursively as follows:

$$x_0 = \frac{b}{\|b\|}$$
$$x_t = \frac{Ax_{t-1}}{\|Ax_{t-1}\|}$$

This method goes back more than hundred years to a paper by Müntz in 1913, but continues to find new applications today.

8.3 Applying Chebyshev polynomials

As we would expect from the development for quadratics, we can use Chebyshev polynomials to get a better solution the polynomial approximation problem that we posed above. The idea is exactly the same with the small difference that we normalize our Chebyshev polynomial slightly differently. This time around, we want to ensure that $p(\lambda_1) = 1$ so that we are picking out the first eigenvalue with the correct scaling.

Lemma 8.4. *A suitably rescaled degree t Chebyshev polynomial achieves*

$$\min_{p(\lambda_1)=1} \max_{\lambda \in [\lambda_2, \lambda_n]} p(\lambda) \leqslant \frac{2}{(1 + \max\{\sqrt{\epsilon}, \epsilon\})^t}$$

where $\epsilon = \frac{\lambda_1}{\lambda_2} - 1$ *quantifies the gap between the first and second eigenvalue.*

Note that the bound is much better than the previous one when ϵ is small. In the case of quadratics, the relevant "ϵ-value" was the inverse condition number. For eigenvalues, this turns into the gap between the first and second eigenvalue.

	$Ax = b$	$Ax = \lambda x$
ϵ	$\frac{1}{\kappa} = \frac{\alpha}{\beta}$	$\frac{\lambda_1}{\lambda_2} - 1$

As we saw before, Chebyshev polynomials satisfy a recurrence relation that can be used to derive an iterative method achieving the bound above. The main shortcoming of this method is that it needs information about the location of the first and second eigenvalue. Instead of describing this algorithm, we move on to an algorithm that works without any such information.

8.4 Conjugate gradient method

At this point, we switch back to linear equations $Ax = b$ for a symmetric positive definite matrix $A \in \mathbb{R}^{n \times n}$. The method we'll see is called *conjugate gradient* and is an

important algorithm for solving linear equations. Its eigenvalue analog is the Lanczos method. While the ideas behind these methods are similar, the case of linear equations is a bit more intuitive.

Definition 8.5 (Conjugate gradient method)**.** We want to solve $Ax = b$, with $A \succ 0$ symmetric. The conjugate gradient method maintains a sequence of three points:

$$\begin{aligned} x_0 &= 0 && (\text{“candidate solution”}) \\ r_0 &= b && (\text{“residual”}) \\ p_0 &= r_0 && (\text{“search direction”}) \end{aligned}$$

For $t \geqslant 1$:

$$\begin{aligned} \eta_t &= \frac{\|r_t\|}{\langle p_{t-1}, Ap_{t-1}\rangle} && (\text{“step size”}) \\ x_t &= x_{t-1} + \eta_t p_{t-1} \\ r_t &= r_{t-1} - \eta_t A r_{t-1} \\ p_t &= r_t + \frac{\|r_t\|^2}{\|r_{t-1}\|^2} p_{t-1} \end{aligned}$$

Lemma 8.6. *The following three equations must always be true for the conjugate gradient method algorithm:*

- $\text{span}(\{r_0, \ldots r_{t-1}\}) = K_t(A, b)$
- *For $j < t$ we have $\langle r_t, r_j\rangle = 0$ and in particular $r_t \perp K_t(A, b)$.*
- *The search directions are conjugate $p_i^\top A p_j = 0$ for $i \neq j$.*

Proof. Proof by induction (see Trefethen and Bau). Show that the conditions are true initially and stay true when the update rule is applied. ■

Lemma 8.7. *Let $\|u\|_A = \sqrt{u^\top Au}$ and $\langle u, v\rangle_A = u^\top Av$ and $e_t = x^* - x_t$. Then e_t minimizes $\|x^* - x\|_A$ over all vectors $x \in K_{t-1}$.*

Proof. We know that $x_t \in K_t$. Let $x \in K_t$ and define $x = x_t - \delta$. Then, $e = x^* - x = e_t + \delta$. We compute the error in the A norm:

$$\begin{aligned} \|x^* - x\|_A^2 &= (e_t + \delta)^\top A(e_t + \delta) \\ &= e_t^\top Ae_t + \delta^\top A\delta + 2e_t^\top A\delta \end{aligned}$$

Bby definition $e_t^\top A = r_t$. Note that $\delta \in K_t$. By Lemma 8.6, we have that $r_t \perp K_t(A, b)$. Therefore, $2e_t^\top A\delta = 0$ and hence,

$$\|e\|_A^2 = \|x^* - x\|_A^2 = e_t^\top Ae_t + \delta^\top A\delta \geqslant \|e_t\|_A .$$

In the last step we used that $A \succ 0$. ■

What the lemma shows, in essence, is that conjugate gradient solves the polynomial approximation problem:

$$\min_{p\colon\ \deg(p)\leqslant t, p(0)=1} \|p(A)e_0\|_A\,.$$

Moreover, it's not hard to show that

$$\min_{p\colon\ \deg(p)\leqslant t, p(0)=1} \frac{\|p(A)e_0\|_A}{\|e_0\|_A} \leqslant \min_{p\colon\ \deg(p)\leqslant t, p(0)=1} \max_{\lambda\in\Lambda(A)} |p(\lambda)|\,.$$

In other words, the error achieved by conjugate gradient is no worse that the error of the polynomial approximation on the RHS, which was solved by the Chebyshev approximation. From here it follows that conjugate gradient must converge at least as fast in $\|\cdot\|_A$-norm than Chebyshev iteration.

References

[TD97] Lloyd N. Trefethen and David Bau, III. *Numerical Linear Algebra*. SIAM, 1997.