princeton univ. F'18 cos 521: Advanced Algorithm Design

Lecture 14: The Power Method and Spectral Methods for Graph Partitioning

Lecturer: *Christopher Musco*

1 The Singular Value Decomposition

Last lecture we proved that any matrix has a singular value decomposition:

Theorem 1 (Singular Value Decomposition (SVD))**.** *Consider $A \in \mathbb{R}^{n\times d}$ and let $r = \min(d,n)$. A can always be written as the product of three matrices, $A = U\Sigma V^T$, where:*

- *$U \in \mathbb{R}^{n\times r}$ is a matrix with orthonormal columns,*
- $\Sigma = \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \end{bmatrix}$ *is a non-negative diagonal matrix with entries $\sigma_1 \geq \ldots \geq \sigma_r \geq 0$,*
- *$V \in \mathbb{R}^{d\times r}$ is a matrix with orthonormal columns.*

U's columns are called the "left singular vectors" of A, V's columns are its "right singular vectors", and $\sigma_1, \ldots, \sigma_r$ are its "singular values".

We also proved the following very useful theorem:

Claim 2 (Truncated SVD)**.** *For any $k \in 1, \ldots, \min(n,d)$, let $U_k \in \mathbb{R}^{n\times k}$ contain the first k columns of U, let $V_k \in \mathbb{R}^{d\times k}$ contain the first k columns of V, and let Σ_k be a $k \times k$ diagonal matrix containing A's first singular values. Then:*

$$\|A - U_k\Sigma_k V_k^T\|_F^2 = \min_{B\in\mathbb{R}^{d\times k}, C\in\mathbb{R}^{k\times n}} \|A - BC\|_F^2.$$

In other words, there is no better rank k approximation for A than $U_k\Sigma_k V_k^T$.

Note that $U_k\Sigma_k V_k^T = AV_kV_k^T$. $V_kV_k^T$ is a projection matrix, so this is a projection of A's rows onto the span of V_k. For any orthonormal matrix $Z \in \mathbb{R}^{d\times k}$, by the matrix Pythagorean theorem, $\|A - AZZ^T\|_F^2 = \|A\|_F^2 - \|AZZ^T\|_F^2$. So $Z = V_k$ can also be said to maximize $\|AZZ^T\|_F^2$ among all Z. Similarly, $\|U_kU_k^TA\|_F^2 \geq \|QQ^TA\|_F^2$ for any matrix $Q \in \mathbb{R}^{n\times k}$ with orthonormal columns.

The SVD gives optimal low-rank approximations for other norms. One useful example is the spectral norm, $\|M\|_2 = \max_{x,\ \|x\|_2=1} \|Mx\|_2$. Try proving the following:

$$\|A - U_k\Sigma_k V_k^T\|_2^2 = \sigma_{k+1}^2 = \min_{B\in\mathbb{R}^{d\times k}, B\in\mathbb{R}^{k\times n}} \|A - BC\|_2^2.$$

2 Connection to Other Matrix Decompositions

The singular value decomposition is closely related to other matrix decompositions:

Eigendecomposition The left singular vectors of A are eigenvalues of $AA^T = U\Sigma^2U^T$ and the right singular vectors are eigenvectors of A^TA. To see that this is the case, note that:

$$AA^Tu_i = U\Sigma V^T V U^T u_i = U\Sigma e_i = \sigma_i u_i.$$

Here e_i is the i^{th} standard basis vector: $U^Tu_i = e_i$ because u_i is orthogonal to all other columns in U.

The connection with eigendecomposition means that any algorithm for eigendecomposition can be used to compute an SVD. Suppose $d \leq n$. Then we can compute A^TA, from which we can compute V using an eigendecomposition algorithm. We then have $\Sigma U^T = AV^T$, so we can obtain Σ and U by normalizing the columns of this matrix and setting σ_i to be the normalization factor for column i. This procedure takes $O(nd^2)$ time to compute A^TA and roughly $O(d^3)$ time to compute the eigendecomposition of this matrix[1]

On another note, you may recall that any real symmetric matrix M has eigendecomposition $U\Lambda U^T$ where U is orthonormal. Λ can have negative diagonal elements, so at least up to changing signs, M's singular vectors are the same as its eigenvectors. It's singular values are the absolute values of its eigenvalues.

Principal Component Analysis (PCA) PCA is almost the same as the SVD, however, before computing singular vectors, we mean center A's rows: $a_i \to a_i - \frac{1}{n}\sum_{j=1}^n a_j$. The right singular vectors of the resulting matrix are call the "principal components" of A.

3 The Power Method

For an $n \times d$ matrix with $n \leq d$, we cannot hope to do much better than $O(nd^2)$ time for computing an SVD. In theory, we can speed up the computation of A^TA and the eigendecomposition of this $n \times n$ matrix with fact matrix multiplication. Doing so achieves a runtime of $O(nd^{\omega-1})$, where ω is the current best known exponent for $d \times d$ matrix multiplication ($\omega = 2.3728639...$ as of 2014 [1]). In practice, however, runtime still scales as $O(nd^2)$.

We want something faster. We are especially interested in algorithms that run more quickly when we only want to compute a few of A's top singular vectors, not all n of them (as is often the case in applications). One such algorithm is the well known *power method*. We present a version below for approximately computing the top right singular vector of A, which can be used to find a best rank 1 approximation:

Power Method

- Initialize $z_0 \in \mathbb{R}^d$ to have every entry a random Gaussian variable. Set $z_0 = z_0/\|z_0\|_2$.

[1] We say roughly "roughly" because technically there is no "exact" algorithm for the SVD, even in the Real RAM model of computation. This is consequence of the Abel-Ruffini theorem. Thus, all SVD algorithms are technically approximation algorithms. However, standard methods obtain *very* good ϵ dependence. E.g. the QR algorithm can compute a factorization $V\Sigma^2V^T$ with $\|V\Sigma^2V^T - A^TA\| \leq \epsilon$ in $O(d^3 + d^2 \log\log(1/\epsilon))$ time. The second term is ignored because it is always lower order in practice.

- Repeat: $z_{t+1} \leftarrow A^T(Az_t)$. $z_{t+1} \leftarrow z_{t+1}/\|z_{t+1}\|_2$.

Theorem 3. *Let $\gamma = \frac{\sigma_1 - \sigma_2}{\sigma_1}$ be a parameter that measures the "gap" between A's first and second singular values. After $t = O\left(\frac{\log(d/\epsilon)}{\gamma}\right)$ iterations, $\|v_1 - z_t\|_2^2 \leq \epsilon$. I.e. z_t is a very good approximate top right singular vector. The power method runs in $O(t \cdot nd)$ time.*

Proof. Write $z_0 = \sum_{i=1}^d \alpha_i v_i$ where v_i is the i^{th} right singular vector of A. Each α_i represents "how much" of singular vector v_i is in z_0. Let $\alpha \in \mathbb{R}^d$ be the vector containing these values. $\alpha = V^T g/\|g\|_2$ where g is a vector of independent Gaussians. By rotational invariance of the Gaussian distribution, $V^T g$ is also a random Gaussian vector. So at least to start, z_0 contains a random amount of every right singular vector in A.

It's not hard to check that $\alpha_1 > 1/\text{poly}(d)$ with high probability and, since z_0 has unit norm, $\max_i \alpha_i = 1$. So we at least have a non-negligible component of v_1 in z_0.

The idea behind the power method is to boost this component so that, eventually, z_t is made up almost entirely of v_1. This is accomplished by repeatedly multiplying by $A^T A$. After t steps, $z_t = c\left(A^T A\right)^t z_0$ for some scale factor c. Since $A^T A = V\Sigma^2 V^T$, after iteration t we have:

$$z_t = \sum_{i=1}^d w_i v_i$$

where each $w_i \sim \sigma_i^{2t}\alpha_i$. By our "gap" assumption, $\frac{\sigma_1}{\sigma_j} \geq 1 + \gamma$ for all $j \geq 2$. Accordingly, after t steps, for all $j \geq 2$,

$$\frac{w_j}{w_1} \leq (1+\gamma)^{2t} \cdot \frac{\alpha_i}{\alpha_1} \leq (1+\gamma)^{2t} \cdot \text{poly}(d).$$

If we set $t = O\left(\frac{\log(d/\epsilon)}{\gamma}\right)$ then we have $\frac{w_j}{w_1} \leq \sqrt{\epsilon/d}$, which means that $w_j \leq \sqrt{\epsilon/2d}$. Since $\|z\|_t = \sum_{j=1}^d w_j^2 = 1$, it follows that $w_1 \geq 1 - \epsilon/2$ and thus $z_t^T v_1 \geq (1 - \epsilon/2)$. So:

$$\|v_1 - z_t\|_2^2 = \|v_1\|_2^2 + \|z_1\|_2^2 - 2z_t^T v_1 \leq \epsilon.$$

□

So when γ is considered constant, power method converges in $\log(d/\epsilon)$ iterations. Accordingly, we can compute a good approximation to the top right singular vector in time $O(nd\log(d/\epsilon))$.

How about when γ is very small? In the most extreme case, when $\gamma = 0$, power method will never converge on v_1 and in fact the dependence on $1/\gamma$ is unavoidable. However, if γ is small, we don't typically care about finding v_1! Since $\sigma_1 = \sigma_2$, v_2 is just as "good" of an eigenvector as v_1. It's a good exercise to prove the following:

Theorem 4. *After $t = O\left(\frac{\log(d/\epsilon)}{\epsilon}\right)$ iterations of power method, z_t satisfies:*

- $\|Az_t\|_2 \geq (1-\epsilon)\sigma_1$
- $\|A - Az_t z_t^T\|_F \leq (1+\epsilon)\|A - Av_1 v_1^T\|_F$

- $\|A - Az_tz_t^T\|_2 \leq (1+\epsilon)\|A - Av_1v_1^T\|_2$

In other words, after $O\left(\frac{\log(d/\epsilon)}{\epsilon}\right)$, by most common measures, projecting rows to z_t still gives a nearly optimal low-rank approximation for A. We've traded a $1/\gamma$ dependence for a $1/\epsilon$ dependence and a different, but arguably more natural approximation guarantee.

4 Beyond Power Method

Last class we discussed how low-rank approximations can be computed in a "greedy" way – i.e. we find a rank 1 approximation to A, substract it off, then find a rank 1 approximation to the remainder, continuing for k steps. We sum up all of these rank-1 approximations to find a rank k approximation. This process is called "deflation" and it's possible to show that it works even when our rank-1 approximations are computed approximately (e.g. with power method).

Other ways of obtaining rank k approximations include "blocked" versions of the power method, where we derive k singular vectors from $\left(A^TA\right)^t Z$ where $Z \in \mathbb{R}^{d\times k}$ is a random Gaussian matrix (instead of just a vector).

In either case, these iterative methods take $O(t \cdot ndk)$ time to compute a nearly optimal rank-k approximation, where either $t = O(\frac{\log d}{\epsilon})$ or depends on gaps between A's singular vectors. In practice, this is typically much faster than computing a full SVD. As an added advantage, all of this runtime complexity comes from matrix-vector multiplications of the form A^TAx, which can be speed up beyond $O(nd)$ time when A is sparse or when parallel processing is available.

Finally, I'll mention that it is actually possible to improve the iteration complexity of the power method to $t = O(\frac{\log d}{\sqrt{\epsilon}})$ using what is known as the Lanczos method. Variations on the Lanczos method are used almost everywhere in practice (e.g. if you run *svds* in MATLAB, Python, etc.). If you are interested, Chapter 10 in [2] gives a relatively simple analysis for the rank-1 case.

5 Matrix decomposition and graphs

In general, algorithms based on singular value decomposition or eigendecomposition are referred to as "spectral methods" – the singular values $\sigma_1, \ldots, \sigma_r$ of a matrix or the eigenvalues $\lambda_1, \ldots, \lambda_r$ are referred to as the "spectrum" of the matrix.

Beyond statistics, data analysis, and machine learning, spectral methods have been very important in developing faster algorithms for graphs, including for classic problems like minimum cut and max flow. Today we will see one particularly nice application.

A big reason for the connection between graphs and matrix decompositions is that the eigenvectors/singular vectors of certain matrix representations of a graph G contain a lot of information about *cuts* in the graph.

Let $G = (V, E)$ be an undirected graph on n nodes. Recall that G's adjacency matrix

A is defined by:

$$A_{u,v} = \begin{cases} 1 & \text{if } (u,v) \in E \\ 0 & \text{if } (u,v) \notin E \end{cases}$$

A is a symmetric matrix, so it has an eigendecomposition $U\Lambda U^T$ where U is orthonormal. For a given eigenvector u_i and corresponding eigenvalue λ_i, $u_i^T A u_i = \lambda_i$.

Consider a vector $z \in \{-1, 1\}^n$. It's not hard to check that $z^T A z$ equals:

$$z^T A z = \sum_{u,v \in V} \mathbb{1}[(u,v) \in E] z_u z_v.$$

Think of z as an indicator vector for a cut between two partitions of vertices, S and T. I.e. $z_u = 1$ for $u \in S$ and $z_u = -1$ for $u \in T$. Every edge within S or T adds a value of 1 to $z^T A z$, which every edge between the partitions adds -1. So, in general, $z^T A z$ will be larger when z is an indicator for "good" partition of G that clusters the graph into two groups of well connected vertices.

In particular, this means that z correlates well with the top eigenvectors of A, which means that these eigenvectors are often useful in finding such cuts.

6 Planted Bisection/Stochastic Block Model/Community Detection

Unfortunately, most optimization problems involving balanced/sparse cuts are NP-hard, but there are many natural "average case" problems to study, which can justify why spectral methods work well in practice. Consider the follow:

Definition 1 (Stochastic Block Model)**.** *Let $G(V, E)$ be a random graph with vertices $V = 1, \ldots, n$. Let S, T form a bisection of V. I.e. $S, T \subset V$ with $S \cup T = V$, $S \cap T = \emptyset$ and $|S| = |T| = n/2$. For probabilities $p > q$, construct G by adding edge (i, j) independently with probability Y_{ij}, where:*

$$Y_{ij} = \begin{cases} p & \textit{if both } i,j \in T \textit{ or } i,j \in S \\ q & \textit{if } i \in T, j \in S \textit{ or } i \in S, j \in T. \end{cases}$$

We can think of S and T as disjoint "communities" in our graph. Nodes are connected randomly, but it is more likely that they are connected to members of their community than members outside their community.

Our goal is to design a spectral method to recover these underlying communities. Today we are just going to a give a sketch of an algorithm/proof.

Let's introduce another matrix $B \in \mathbb{R}^{n \times n}$ defined as follows:

$$B_{ij} = \begin{cases} p & \text{if } i,j \in T \text{ or } i,j \in S \\ q & \text{if } i \in T, j \in S \text{ or } i \in S, j \in T. \end{cases}$$

It is not hard to see that $B = \mathbb{E}[A] + pI$, where I is an $n \times n$ identity. Accordingly, at least in expectation, A has the eigenvectors as B. What are these eigenvectors?

B is rank two, so it only has two, u_1 and u_2, where:

$$u_1(i) = \frac{1}{\sqrt{n}} 1 \ \forall i,$$

$$u_2(i) = \begin{cases} \frac{1}{\sqrt{n}} 1 & \forall i \in S \\ \frac{1}{\sqrt{n}} - 1 & \forall i \in T. \end{cases}$$

$Bu_1 = \frac{n}{2}(p+q)u_1$ and $Bu_2 = \frac{n}{2}(p-q)u_2$. In this case, u_1 and u_2 are also B's singular vectors.

So, if we could compute B's eigenvectors, we could immediately recover our community by simply examining u_2. Of course, we don't have access to B, but we do have accesses to a perturbed version of the matrix via:

$$\hat{A} = A + pI.$$

Consider $R = B - \hat{A}$. Classic perturbation theory results in linear algebra tell us that if $\|R\|_2$ is small, then $\hat{A}$'s eigenvalues and eigenvectors will be close to those of B.

Let B have eigenvectors $u_1, \ldots, u_n$ and eigenvalues $\lambda_1, \ldots, \lambda_n$. Let $\hat{A}$ have eigenvectors $\hat{u}_1, \ldots \hat{u}_n$ and eigenvalues $\hat{\lambda}_1, \ldots, \hat{\lambda}_n$. Using ideas from the past few lecture you could already prove the following result, which is a good exercise:

Claim 5. *If B and $\hat{A}$ are real symmetric matrices with $\|B - \hat{A}\|_2 \leq \epsilon$, $\forall i$,*

$$|\lambda_i - \hat{\lambda}_i| \leq \epsilon.$$

In words, if $\hat{A}$ and B are close in spectral norm, their eigenvalues are close. For our application, we further need that the matrices *eigenvectors are close.* Below is a classic result quantifying this – you can find a simple proof of a slightly weaker version in [3].

Claim 6 (Davis-Kahan, 1970 [4])**.** *Suppose B and $\hat{A}$ are real symmetric matrices with $\|B - \hat{A}\|_2 \leq \epsilon$. Let θ_i denote the angle between u_i and $\hat{u}_i$. For all i,*

$$\sin\theta_i \leq \frac{\epsilon}{\min_{j \neq i} |\lambda_i - \lambda_j|}.$$

Let's unpack this claim. It says that if B and $\hat{A}$ are close in spectral norm, then their corresponding eigenvectors are close. However, the distance is effected by a factor of $1/|\lambda_i - \lambda_j|$. This makes sense – suppose $\lambda_i < \lambda_{i+1} + \epsilon$. Then a pertubation with spectral norm ϵ could cause the u_i and u_{i+1} to "swap" order – specifically just add $\epsilon u_{i+1} u_{i+1}^T$ to B to cause such a change. In the perturbed matrix, $\hat{u}_i = u_{i+1}$, which is orthogonal to u_i.

Fortunately, in our case, we have a gap between B's eigenvalues – in particular, $|\lambda_2 - \lambda_1| \geq nq$ and $|\lambda_2 - 0| = \frac{n}{2}(p-q)$. Let's assume a challenging regime where q is close to p and thus $\frac{n}{2}(p-q) \leq nq$).

A simple corollary of Claim 6 is that $\|u_i - \hat{u}_i\|_2 \leq \frac{\sqrt{2}\epsilon}{\min_{j \neq i} |\lambda_i - \lambda_j|}$.

As an estimate for our community indicator vector u_2, let's consider $\text{sign}(\hat{u}_2)$. Suppose this estimate differs from u_2 on k entries. Then it must be that:

$$\|\hat{u}_2 - \mu_2\|_2 \geq \sqrt{\frac{k}{n}}$$

So, by the eigenvector perturbation argument, we can bound

$$k \leq O\left(\frac{\epsilon^2}{n(p-q)^2}\right)$$

7 Eigenvalues of Random matrices

So we are left to bound $\|R\|_2$. $R = B - \hat{A}$ is a random matrix with half of its entries equal to p with probability $(1-p)$ and $(p-1)$ with probability p, and the other half equal to q with probability $(1-q)$ and $(q-1)$ with probability q.

It is possible to prove:

Theorem 7 (From [5]). *If $p \geq O(\log^4 n/n)$, then with high probability,*

$$\|R\|_2 \leq O(\sqrt{pn})$$

You will prove a very related (but slightly looser statement on the problem set).

With this bound in place, we immediately have that our spectral algorithm recovers the hidden partition with a number of mistakes bounded by:

$$k = O\left(\frac{p}{(p-q)^2}\right).$$

This is very good. Even when $q = p - O(1/\sqrt{n})$ (e.g. our probabilities are very close, so the communities should be hard to distinguish) we only make $O(n)$ mistakes – i.e. we can guess a large constant fraction of the community identities correctly.

References

[1] Le Gall, François. Powers of Tensors and Fast Matrix Multiplication. Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation, 296–303. 2014.

[2] Sushant Sachdeva, Nisheeth K. Vishnoi. Faster Algorithms via Approximation Theory. Foundations and Trends in Theoretical Computer Science. 2013. https://theory.epfl.ch/vishnoi/Publications_files/approx-survey.pdf.

[3] Daniel Spielman. Spectral Partitioning in a Stochastic Block Model. Lecture, Yale University. 2015. http://www.cs.yale.edu/homes/spielman/561/lect21-15.pdf.

[4] Chandler Davis, William Morton Kahan. The rotation of eigenvectors by a perturbation. SIAM Journal on Numerical Analysis, 7(1):146, 1970.

[5] Van Vu. Spectral norm of random matrices. Combinatorica, 27(6):721736, 2007.