Tensor decomposition via sum of squares

One of the most basic tools in data analysis is *least squares regression*. The basic setting is where we are given points $v_1, \ldots, v_m \in \mathbb{R}^2$ and we wish to find the line L that minimizes the sum of square distances of the v_i's to L. More generally, we can think of case that $v_1, \ldots, v_m \in \mathbb{R}^n$ and we wish to find a k dimensional subspace L that minimizes the sum of squares of distances to L. Another way to think about this is that we are given the matrix $M = \sum_{i=1}^m v_i v_i^\top$ and we want to find a rank r matrix $L = \sum_{i=1}^r w_i w_i^\top$ that minimizes $\|M - L\|_F^2$ (where $\|A\|_F = \mathrm{Tr}(AA^\top)^{1/2}$ denotes the *Frobenius norm* of a the matrix A).

This problem can be efficiently solved by doing a *singular value decomposition (SVD)* of the p.s.d. matrix M as $M = \sum_{i=1}^n \lambda_i w_i w_i^\top$ and taking the vectors corresponding to the largest k eigenvalues. However, SVD is not always sufficient to recover information we are interested in it. For example, if the v_i's come from two separate "clusters" centered at w_1 and w_2 respectively, then SVD can recover the subspace $Span\{w_1, w_2\}$ but not the individual centers.

The underlying difficulty comes from the fact that very different configurations of vectors $v_1, \ldots, v_m$ can yield the same second moment matrix $\sum v_i v_i^\top$. It turns out that in many cases one can break this symmetry by looking at the *third* (or higher) moment matrix $\sum_i v_i^{\otimes 3}$. This suggest the following task of *tensor decomposition*:

1. Definition. Let $T \in \mathbb{R}^{n^d}$ be a tensor. A *rank r decomposition of T* is a set of vectors $\{a_{i,j}\}_{i \in [r], j \in [d]}$ such that

$$T = \sum_{i=1}^r a_{i,1} \otimes \cdots \otimes a_{i,d} \tag{1}$$

where for $v \in R^n$ and $w \in \mathbb{R}^m$, $v \otimes w$ is the mn dimensional vector where $(v \otimes w)_{i,j} = v_i w_j$.

The *rank* of a tensor T is the minimum r such that T has a rank r decomposition. Let $\|\cdot\|$ be some norm on $\mathbb{R}^{n^d}$. An *ϵ-approximate rank r decomposition* of T w.r.t. the norm $\|\cdot\|$ is a set of vectors as above such that

$$\|T - \sum_{i=1}^r a_{i,1} \otimes \cdots \otimes a_{i,d}\| \leq \epsilon \|T\| \tag{2}$$

The *tensor decomposition problem* is to recover, given a rank r tensor T, a rank r decomposition of T. *Noisy* tensor decomposition is the task of recovering an *approximate* decomposition of T when such a decomposition exists. Tensor decomposition is an extremely useful

primitive Kolda and Bader [2009] but unfortunately it is NP-hard in general (Hillar and Lim [2013]). However, there are certain conditions under which a decomposition can be recovered, and the sos algorithm turns out to be extremely useful in this setting.

Classical tensor decomposition algorithms

While tensor decomposition is a hard problem in general, there are algorithm that succeed for it in certain regimes. The philosophy behind them is that a la Halevy et al. [2009], a sufficient amount of data can compensate for computational difficulty. Specifically in the context of tensor decomposition the *data* is the tensor $T \in \mathbb{R}^{n^d}$ and so we have a "blessing of dimensionality" where the problem actually becomes *easier* as d grows for a fixed rank r.

Let's restrict attention to the *symmetric* noiseless case, where the tensor T has a decomposition of the form $T = \sum_{i=1}^{r} a_i^{\otimes d}$. Note that $a_1, \ldots, a_r$ can be described using rn numbers, and so one might hope to be able to recover these from the n^d dimensional tensor as long as $r \ll n^{d-1}$. Indeed, under certain natural conditions this would be possible information theoretically. However, known efficient algorithms require (in addition to other conditions) the rank r to be much smaller.

A classical algorithm, attributed to Jennrich (Harshman [1970],Leurgans et al. [1993]) can recover the decomposition in the case that $d = 3$ and the vectors are $a_1, \ldots, a_r$ are *linearly independent* (and so in particular $r < n$). It works as follows:

2. Algorithm (Jennrich's tensor decomposition). The input is a 3-tensor $T \in \mathbb{R}^{n^3}$ such that $T = \sum_{i=1}^{r} a_i^{\otimes 3}$ for linearly independent $a_1, \ldots, a_r$. For simplicity assume that the a_i's are orthogonal to one another. (We will see how to relax this assumption later.) Compute a decomposition of T as follows:

1. Pick $v \in \mathbb{R}^n$ at random, and compute $M = Tv$, where we think of T here as an $n^2 \times n$ matrix.
2. Do a singular value decomposition of $M = \sum_{i=1}^{r'} \lambda_i w_i w_i^\top$.
3. Solve a least squares problem to find the $\alpha_1, \ldots, \alpha_r$ that minimize $\|T - \sum_{i=1}^{r} \alpha_i w_i^{\otimes 3}\|_2^2$. (Note that the only unknown here are the variables $\alpha_1, \ldots, \alpha_r$.)

To see that this algorithm works, note that

$$M = \sum_{i=1}^{r} \langle a_i, v \rangle a_i a_i^\top . \tag{3}$$

and so since the a_i's are orthogonal this is going to be a singular value decomposition. Moreover, since v as random the values $\langle a_i, v \rangle$ are going to be distinct with probability one, and hence this singular value decomposition is unique. Thus if we run SVD we will get vectors $w_1, \ldots, w_r$ that are (up to permuting the indices) equal up to scale to $a_1, \ldots, a_r$ and hence there would be a solution to the $\alpha_1, \ldots, \alpha_r$ that makes $T = \sum_{i=1}^{r} \alpha_i w_i^{\otimes 3}$. (Note that in practice we need to worry about highly non-trivial issues of noise and numerical stability, but we ignore these at the moment for the purposes of the current discussion.)

Whitening transformation: Suppose that the a_i's are linearly independent but not orthogonal to one another. If we are given the second moment matrix $M_2 = \sum_{i=1}^{r} a_i a_i^\top$ then by applying the transformation $M_2^{-1/2}$ (restricted to the image space of M_2 if $r < n$) we can reduce to the orthogonal case.

Recovering higher rank tensors: One limitation of Jennrich's algorithm is that it only works in the case that the rank r of the tensor is at most n, while often we are interested in the so called "overcomplete" case when $r \gg n$. Indeed, as mentioned above, one could hope to recover a rank r decomposition of a d-dimensional tensor as long as $r \ll n^{d-1}$. Jennrich's algorithm can scale to higher rank, at the expense of larger dimension. For example, if r is some constant times n^2 and the vectors are "generic" then we would expect the vectors $a_1^{\otimes 2}, \ldots, a_r^{\otimes 2}$ to be linearly independent, which means we can run Jennrich's algorithm on the tensor $\sum a_i^{\otimes 6}$, thinking of it as a 3-tensor over n^2. A closer examination shows that a dimension 5 tensor suffices (can you see why?). More generally, we can get an algorithm for a rank $\Omega(n^{\lceil d/2 \rceil + 1})$ decomposition of generic d-tensors using Jennrich's algorithm. The best sos based algorithm achieve rank $\Omega(n^{d/2})$ which (depending on whether d is odd or even) saves a multiplicative factor of about n or n^2 in the number of observations. This multiplicative factor is non-trivial, but it is not the only advantage of these sos-based algorithms. They also enjoy better robustness to *noise*, which can be even more crucial than saving data for certain applications.

The "brute data algorithm"

We saw that if we have enough data (i.e., about r^3 observations instead of the minimum of $O(rn)$), we can use Jennrich's algorithm to recover the decomposition. It turns out that if we have even more data, we can use a simpler algorithm:[1]

[1] This simpler algorithm is not more efficient, but it has a conceptually simpler and more robust analysis, which will be useful for us later on.

3. Lemma. *Suppose that $d > O(1/\epsilon^2)\log r$ and $T = \sum_{i=1}^r a_i^{\otimes d}$ and that $\|a_i\| = 1$ for all i. Then for every i, with probability at least $1/n^{O(1/\epsilon^2)}$ over a Gaussian $v \in \mathbb{R}^n$,*

$$a_i^{\otimes 2} T v^{\otimes d-2} \geq (1-\epsilon)\|a_i^{\otimes 2}\|\|Tv^{\otimes d-2}\| \,, \tag{4}$$

treating T as an $n^2 \times n^{d-2}$ matrix.

Lemma 3 yields an algorithm that can iteratively learn the a_i's vectors one by one, since if we choose a random v then with high probability the matrix $Tv^{\otimes d-2}$ will be close to a rank one matrix of the form $aa^\top$ where a is (up to scaling) equal to one of the a_i's.

Proof. For any i_0, with probability at least $n^{-O(1/\epsilon^2)}$, we will get that $(*)$ $\langle a_{i_0}, v\rangle = \alpha$ for $\alpha \geq (100/\epsilon)\sqrt{\log n}$ standard deviations. If we condition on this event, then the distribution of $\langle u, v\rangle$ for $u \perp a_{i_0}$ is a standard Gaussian and so by the union bound with high probability in this conditional space, $(**)$ for every unit vector u orthogonal to a_{i_0}, $\langle a_{i_0}, v\rangle$ is at most $2\sqrt{\log n} = (\epsilon/50)\alpha$ standard deviations.

Suppose that that both $(*)$ and $(**)$ happen. Then, for every vector $a = \sqrt{1-\epsilon^2}a_i + \epsilon u$,

$$\langle a, v\rangle \leq \alpha\sqrt{1-\epsilon^2} + \epsilon(\epsilon/50)\alpha \leq (1-\epsilon^2/3)\alpha \tag{5}$$

And hence in particular if $\langle a_i, a\rangle \leq 1-\epsilon^2$ then when this event occurs then $\langle a_i, v\rangle^{d-2} \ll \langle a_i, v\rangle/r$. Thus, in the tensor $Tv^{\otimes d-2} = \sum_{i=1}^r \langle a_i, v\rangle^{d-2} a_i^2$ the contributions of a_i's where $\langle a_i, a_{i_0}\rangle \leq 1-\epsilon^2$ are negligible compared to the contributions of the a_i's (of which there is at least one, namely i_0) where $\langle a_i, a_{i_0}\rangle \geq 1-\epsilon^2$. This completes the proof. □

Dreaming about moments

We have seen that in the context of tensor decomposition, we can trade *time* for *data*. While we don't know of subexponential algorithms that can recover a decomposition from the information theoretically minimal number of moments (e.g., rank $\Omega(n^{d-1})$ for

d-tensors), we can get faster algorithms if we have access to more observations. Somewhat surpisingly, the sos algorithm allows us to make the reverse tradeoff. That is, it allows us to use computation time to compensate for the lack of observations. The idea is that we use our pseudo-distributions to produce "fake moments" of higher degree than the ones we've truly observed. We then run a simple algorithm such as Jennrich's or the "brute data algorithm" on the fake moments. As long as the *analysis* of these simple algorithms can be embedded in the low degree sos proof system then we can show that recovery will still succeed even if they are given *fake* moments. One can think of this approach as "dreaming" of moments that we don't have, and then running the algorithm on those "illusionary" moments and bringing back the solution from the dream.

Applying this methodology to the "brute data algorithm", we obtain the following tensor decomposition algorithm (Barak et al. [2015]).[2]

[2] The algorithm makes sense for 3-tensors, but the analysis is somewhat simpler for 4 tensors, so we describe it in this setting here.

4. Algorithm (dream of brute data). Given input $T \in \mathbb{R}^{n^4} = \sum_{i=1}^{r} a_i^{\otimes 4}$, operate as follows:

1. Use the sum-of-squares algorithm to compute a degree $d = O(\log n/\epsilon^2)$ pseudo-distribution μ such that $\tilde{\mathbb{E}}_{\mu(a)} a^{\otimes 3} = \frac{1}{r}T$.
2. Compute $\hat{T} = \tilde{\mathbb{E}}_{\mu(a)} a^{\otimes d}$ and apply the "brute data algorithm" on it:

- Pick v to be a random Gaussian.
- Pick w using the quadratic sampling lemma from a distribution matching $\tilde{\mathbb{E}}_{\mu(a)} \langle v, a\rangle^{d-2} a^{\otimes 2}$.[3]
- Output w

[3] Note that if $M = \tilde{\mathbb{E}}_{\mu(a)} \langle v, a^{d-2}\rangle a^{\otimes 2}$ is close to a rank one matrix $uu^\top$ then the vector w we output will be highly correlated with u.

This algorithm outputs a single vector which we can repeat to get a tensor decomposition. Thus showing that this algorithm succeeds in quasipolynomial time boils down to the following theorem:

5. Theorem. *Suppose that $r = n^{2-\epsilon}$, then there is $d = O(\log n/\epsilon^2)$ such that if a_i's are random Gaussian vectors then with probability $1/qpoly(n)$, the vector w that is output satisfies $\langle w, a_1\rangle^2 \geq 0.99\|w\|\|a_1\|$.*

Proof. The main property we will use about random Gaussian vectors $a_1, \ldots, a_r$, for $r \ll n^2$, is that with high probability... □

Making it more efficient: This algorithm runs in quasipolynomial time, but Ma et al. [2016] have shown an analog sos algorithm that "dreams" of using Jennrich's algorithm instead of the brute data one,

and can run in polynomial time. Also, Hopkins et al. [2016] used the *analysis* of sos-based tensor decomposition to extract a *direct* tensor decomposition algorithm that does not go through solving general semidefinite programs, and hence is much more efficient.

References

Boaz Barak, Jonathan A. Kelner, and David Steurer. Dictionary learning and tensor decomposition via the sum-of-squares method. In *STOC*, pages 143–151. ACM, 2015.

Alon Halevy, Peter Norvig, and Fernando Pereira. The unreasonable effectiveness of data. *IEEE Intelligent Systems*, 24(2):8–12, 2009.

Richard A Harshman. Foundations of the parafac procedure: Models and conditions for an" explanatory" multi-modal factor analysis. 1970.

Christopher J. Hillar and Lek-Heng Lim. Most tensor problems are NP-hard. *J. ACM*, 60(6):Art. 45, 39, 2013. ISSN 0004-5411. doi: 10.1145/2512329. URL http://dx.doi.org/10.1145/2512329.

Samuel B. Hopkins, Tselil Schramm, Jonathan Shi, and David Steurer. Fast spectral algorithms from sum-of-squares proofs: tensor decomposition and planted sparse vectors. In *STOC*, pages 178–191. ACM, 2016.

Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

S. E. Leurgans, R. T. Ross, and R. B. Abel. A decomposition for three-way arrays. *SIAM J. Matrix Anal. Appl.*, 14(4):1064–1083, 1993. ISSN 0895-4798. doi: 10.1137/0614071. URL http://dx.doi.org/10.1137/0614071.

Tengyu Ma, Jonathan Shi, and David Steurer. Polynomial-time tensor decompositions with sum-of-squares. *arXiv preprint arXiv:1610.01980*, 2016.