CS 2243 Fall 2024 Scribe: Sara Fish
September 9 Based on notes by: Kevin Luo, Oam Patel, and Benjamin Schiffer

Lecture 2: Tensor Decomposition, Jenrrich's Algorithm, and Applications

1 Motivation

This lecture falls under the theme of proving upper bounds. In the first part of this course, we will focus on building an algorithmic toolkit. Later, we will dive into lower bounds and modeling as well.

1.1 Historical motivation

Today we will study algorithms for **factor analysis**. For some motivation, we will consider one of its earliest applications.

Charles Spearman (1863-1945) pioneered factor analysis in psychology. He posited that there are two kinds of intelligence: "mathematical" and "verbal". Let's look at a toy model of this.

Consider n students and m tests. Let M be a $n \times m$ matrix comprising of the students' scores on the tests, so that student i scored M_{ij} on test j. To model two different factors influencing students' test scores, we consider a rank 2 decomposition of M. Specifically, we write $M \approx UV^T$, where U is a $n \times 2$ matrix and V is a $m \times 2$ matrix. Let $a_i \in \mathbb{R}^2$ denote the ith row of U and $b_j \in \mathbb{R}^2$ denote the jth row of V. Then, for each student i and each test j, we obtain the (approximate) score by taking the dot product $\langle a_i, b_j \rangle$.

Now let $u_1, u_2 \in \mathbb{R}^n$ denote the two columns of U, and similarly $v_1, v_2 \in \mathbb{R}^m$ the two columns of V. Then we can write the low-rank decomposition as a sum of outer products: $M = UV^T = \sum_{i=1}^{k} u_i v_i^T$, where k is the rank of the decomposition (in this example $k = 2$). This is an example of **factor analysis**: we have decomposed the students' test scores M into two components, $u_1 v_1^T$ and $u_2 v_2^T$.

However, there is bad news: the decomposition of M into U and V is not uniquely determined. For any orthogonal matrix O, if $M = UV^T$, then we have $M = \tilde{U}\tilde{V}^T$ as well, for $\tilde{U} := UO$ and $\tilde{V} := VO$. Here is a quick calculation demonstrating why:

$$\tilde{U}\tilde{V}^T = UO(VO)^T = UOO^TV^T = UV^T.$$

This is called the **rotation problem**. Many workarounds have been proposed. For example, one workaround is nonnegative matrix factorization, in which we impose the additional constraint that all matrix coefficients are required to be nonnegative.

In this lecture we will study a different approach that will motivate our study of tensor decomposition. Recall our low-rank decomposition $M = \sum_{i=1}^{k} u_i v_i^T$. We can equivalently write this as a sum of tensor products $M = \sum_{i=1}^{k} u_i \otimes v_i$. In other words, we view M as a 2-tensor. Suppose we had a third axis of information about this problem, say, various *experimental conditions* for all student-test pairs, denoted by $w_1, \ldots, w_k$. Then our data could be written as a 3-tensor: $T = \sum_{i=1}^{k} u_i \otimes v_i \otimes w_i$. It turns out that, under mild conditions, with this added dimension, $\{u_i\}_{i=1}^{k}$, $\{v_i\}_{i=1}^{k}$, and $\{w_i\}_{i=1}^{k}$ can be uniquely recovered from T.

In other words, the rotation problem is an artifact of two-dimensional space, and disappears when we consider three or more dimensions.

2 Tensors

Let's start by giving the definition of a tensor. Order 1 tensors are just vectors and order 2 tensors are just matrices. For today, we will also work with order 3 tensors.

Definition 1 (Order 3 tensor)**.** *An order 3 tensor $T \in \mathbb{R}^{r \times s \times t}$ is an array of numbers with entries T_{ijk} for $i \in [r], j \in [s], k \in [t]$.*

There are higher-order analogues of everything we will discuss today, but for simplicity we will focus on order 3 tensors. Next we define the notion of rank for a tensor.

Definition 2 (Tensor rank)**.** *The rank of an order 3 tensor T is the smallest k for which T admits a decomposition of the form*

$$T = \sum_{i=1}^{k} u_i \otimes v_i \otimes w_i.$$

This decomposition of T is also called CP decomposition.

For example, $T = \sum_{i=1}^{k} u_i \otimes v_i \otimes w_i$ is a order 3 tensor. Each of the $u_i \otimes v_i \otimes w_i$ is a rank 1 tensor. As an exercise, try showing that if T is a $d \times d \times d$ tensor, then its rank is at most d^2. This is analogous to how the rank of a $d \times d$ matrix is at most d.

Computational intractability. Many feature of matrices that we take for granted don't hold for tensors: even tractability of basic linear algebraic primitives! For example, it is NP-Hard to compute, or even approximate, the following: best low-rank (or even rank 1) approximation, rank, operator norm, eigenvalues, and eigenvectors [HL09].

2.1 Tensor decomposition

The following algorithm is attributed to Jennrich, but in fact the history behind the name is murky, [Mat].

Theorem 1 ("Jennrich")**.** *Given $T \in \mathbb{R}^{d \times d \times d}$ with a symmetric CP decomposition*

$$T = \sum_{i=1}^{k} u_i^{\otimes 3}$$

for $u_1, \ldots, u_k$ that are linearly independent, there is a poly(d) time algorithm for recovering $u_1, \ldots, u_k$ exactly.

Intuitively, this is saying that low rank tensors can be decomposed if their components are linearly independent. Note this only works for tensors of sufficiently low rank: we must have $k \leq d$ due to the linear independence condition, while in general, a $d \times d \times d$ tensor could have rank up to d^2.

This result continues to hold for non-symmetric tensors under slightly looser conditions:

Theorem 2 ("Jennrich")**.** *Given $T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ of the form*

$$T = \sum_{i=1}^{k} u_i \otimes v_i \otimes w_i$$

where $\{u_i\}$, $\{v_i\}$, $\{w_i\}$ satisfy

1. *$u_1, \ldots, u_k$ are linearly independent*
2. *$v_1, \ldots, v_k$ are linearly independent*
3. *$d_3 \geq 2$ and no two w_i, w_j are collinear (note that $d_3 = 1$ is the matrix case and hence this restriction makes sense, also note that the $\{w_i\}$ need not be linearly independent, but just none can be multiples of another)*

then there exists a poly(d) time algorithm to recover $\{u_i\}, \{v_i\}, \{w_i\}$.

These results show that there is exactly one way of decomposing a low-rank tensor (apart from trivial swap symmetries). Returning to our original motivation: in Spearman's setting, the rotation problem would be solved by measuring the scores under two different experimental conditions (i.e. adding a third axis).

Before we can prove these theorems, we will need to introduce one essential tensor operation: contraction.

2.2 Tensor contraction

Throughout this section, let $T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ be given by $T = \sum_{i=1}^{k} u_i \otimes v_i \otimes w_i$.

First we introduce the notion of **tensor contraction**. Below we define contraction for order 3 tensors, but this naturally extends to higher orders.

Definition 3 (Tensor Contraction)**.** *Let $z_1 \in \mathbb{R}^{d_1}$, $z_2 \in \mathbb{R}^{d_2}$, and $z_3 \in \mathbb{R}^{d_3}$. The* contraction *of $T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ along z_1, z_2, z_3 is defined as*

$$T(z_1, z_2, z_3) := \sum_{a \in [d_1], b \in [d_2], c \in [d_3]}^{d} T_{a,b,c}(z_1)_a (z_2)_b (z_3)_c.$$

One way to understand tensor contraction is to think of it as a tensor analogue of computing the quadratic form of a matrix. Another way to understand tensor contraction is to view it as evaluating a polynomial. Specifically, we can associated each tensor T with a polynomial $p : \mathbb{R}^{d_1 \times d_2 \times d_3} \to \mathbb{R}$ given by

$$p(z_1, z_2, z_3) = \sum_{a,b,c} T_{abc}(z_1)_a (z_2)_b (z_3)_c.$$

Then the operation of tensor contraction is just evaluating this polynomial. This way of understanding tensor contraction will be helpful for next lecture, where we will see an interesting connection between tensor decomposition and polynomial evaluation. For the purposes of this lecture, it suffices to view tensor contraction as an algebraic operation.

What we will need for Jennrich's algorithm will be not just tensor contraction, but **partial contraction**. Tensor contraction takes three inputs, and outputs a number. By contrast, partial contraction might output a number, vector, matrix, or higher order tensor. Today, we will work with partial contractions along a single mode:

Definition 4 (Partial Contraction)**.** *The partial contraction, denoted $T(:, :, z) : \mathbb{R}^{d_3} \to \mathbb{R}^{d_1 \times d_2}$, is given by*

$$T(:, :, z)_{ij} := T(e_i, e_j, z) = \sum_{k=1}^{d_3} T_{ijk} z_k.$$

Partial contraction along a single mode is the tensor analogue of matrix-vector multiplication: for a $d \times d$ matrix M and $z \in \mathbb{R}^d$, we have $M(:, z) = Mz$.

3 Jennrich's Algorithm and Proof of Correctness

3.1 Jennrich's algorithm

For a formal description of Jennrich's algorithm see Algorithm 1. This algorithm, also known as *simultaneous diagonalization*, was not actually the algorithm Jennrich proposed (alternating least squares), as discussed here (also linked in Section 2.1). Instead, this algorithm seems to be drawn from [LRA93].

On a high level, the algorithm uses the same "simultaneous diagonalization" trick as in the matrix pencil method. We sample two random $z, z' \sim S^{d-1}$ and compute $M_z := T(:, :, z)$ and $M_{z'} := T(:, :, z')$. If we view T as a bunch of stacked "layers" of matrices, M_z and $M_{z'}$ are taking two randomly weighed linear combinations of these layers. As in matrix pencil method, our next step is to "divide" these matrices: to consider $M_z M_{z'}^{+}$ and $M_z^{+} M_{z'}$ (where + denotes pseudoinverse). It turns out (see Lemma 1 for full computation) that the eigenvectors of $M_z M_{z'}^{+}$ give us $\{u_i\}_{i=1}^{k}$, and the eigenvectors of $(M_z^{+} M_{z'})^T$ give us $\{v_i\}_{i=1}^{k}$. Finally, to obtain $\{w_i\}_{i=1}^{k}$, notice that each entry of the tensor T is given by $T_{a,b,c} := \sum_{\ell=1}^{d} (u_\ell)_a (v_\ell)_b (w_\ell)_c$. So we have d^3 linear constraints and kd unknowns $(w_\ell)_c$ for $\ell \in [d], c \in [k]$. From these (redundant, and it turns out consistent) constraints we can solve for $\{w_i\}_{i=1}^{k}$ (see Lemma 2 for full computation).

Algorithm 1: JENNRICH(T)

Input: $T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$

Output: Determines $\{u_i\}$, $\{v_i\}$, $\{w_i\}$ such that $T = \sum_i u_i \otimes v_i \otimes w_i$

$z, z' \leftarrow_{i.i.d.} \text{Unif}(S^{d-1})$

$M_z \leftarrow T(:, :, z)$

$M_{z'} \leftarrow T(:, :, z')$

$(\lambda_i, u_i)_{i=1}^d \leftarrow \text{EIGENDECOMPOSE}(M_z M_{z'}^+)$

// A^+ denotes pseudo-inverse; λ_i eigenvalues; u_i corresponding eigenvectors

$(\mu_j, v_j)_{j=1}^d \leftarrow \text{EIGENDECOMPOSE}((M_z^+ M_{z'})^\top)$

// Match u_i and v_i by fact eigenvalues should be reciprocal

$\{(u_i, v_i)\}_{i=1}^k \leftarrow \{(u_i, v_j) \mid \lambda_i \mu_j = 1\}$ // exactly k such pairs

// now we solve for the w's with a linear system

$\boldsymbol{\lambda}_{(a,b),c} = (u_c)_a (v_c)_b$ // $\boldsymbol{\lambda} \leftarrow \mathbb{R}^{d_1 d_2 \times k}$

$\mathbf{T}_{\text{matrix}} = \text{reshape}(T, (d_1 d_2, d_3))$ // $\mathbf{T}_{\text{matrix}} \in \mathbb{R}^{d_1 d_2 \times d_3}$

// Let $\mathbf{W}$ be the matrix with w_i as its i-row, meaning $\mathbf{W} \in \mathbb{R}^{k \times d_3}$ and $\mathbf{T}_{\text{matrix}} = \boldsymbol{\lambda}\mathbf{W}$

$\mathbf{W} = \boldsymbol{\lambda}^+ \mathbf{T}_{\text{matrix}}$

return $\{u_i\}_{i=1}^k, \{v_i\}_{i=1}^k, \{w_i\}_{i=1}^k$

3.2 Proof of Correctness

In this section, we prove the correctness of Jennrich's algorithm (Algorithm 1).

Lemma 1. *In the notation from Algorithm 1,*

$$M_z \cdot M_{z'}^+ = U D_z D_{z'}^{-1} U^+$$

and

$$M_z^+ \cdot M_{z'} = (V^\top)^+ D_z D_{z'}^{-1} V^\top.$$

Proof. First note that

$$\begin{aligned} M_z &= \sum_{i=1}^k (u_i \otimes v_i \otimes w_i)(:, :, z) \\ &= \sum_{i=1}^k u_i \otimes v_i \cdot \langle w_i, z \rangle \\ &= U D_z V^\top. \end{aligned}$$

Likewise, $M_{z'} = U D_{z'} V^\top$, where

$$\begin{array}{cc} U \in \mathbb{R}^{d \times k} & U = [u_1, u_2, \ldots, u_k] \\ V \in \mathbb{R}^{d \times k} & V = [v_1, v_2, \ldots, v_k] \\ D_z \in \mathbb{R}^{k \times k} & D_z = \text{diag}(\langle w_1, z \rangle, \ldots, \langle w_k, z \rangle) \\ D_{z'} \in \mathbb{R}^{k \times k} & D_{z'} = \text{diag}(\langle w_1, z' \rangle, \ldots, \langle w_k, z' \rangle) \end{array}.$$

Using this representation, we get that

$$\begin{aligned} M_z \cdot M_{z'}^+ &= U D_z V^\top (U D_{z'} V^\top)^+ \\ &= U D_z V^\top (V^\top)^+ D_{z'}^{-1} U^+ \\ &= U D_z ((V)^+ V)^\top D_{z'}^{-1} U^+ \\ &= U D_z D_{z'}^{-1} U^+ \end{aligned}$$

where the final equality comes from the fact that V has linearly independent columns[1]. The same holds for $M_z^+ \cdot M_{z'}$ by a symmetric argument.

□

Lemma 1 shows that $M_z M_{z'}^+$ admits a diagonalization, and in particular, its eigenvectors with nonzero eigenvalue are exactly $\{u_i\}_{i=1}^k$ (the columns of U), where u_i has eigenvalue $\frac{\langle w_i, z\rangle}{\langle w_i, z'\rangle}$. Similarly, Lemma 1 shows that the eigenvectors with nonzero eigenvalue of $(M_z^+ M_{z'})^\top$ are $\{v_i\}_{i=1}^k$ (the columns of V), where v_i has eigenvalue $\frac{\langle w_i, z'\rangle}{\langle w_i, z\rangle}$.

Thus, by calculating the eigendecomposition of $M_z M_{z'}^+$ and $(M_z^+ M_{z'})^\top$, we obtain $\{u_i\}_{i=1}^k$ and $\{v_i\}_{j=1}^k$ up to permutation. We can then pair up them up appropriately by using the fact that the corresponding eigenvalues of u_i and v_i are reciprocals of one another. Note that the non collinearity condition of the w_i necessitates that there will be no duplicate nonzero eigenvalues.

Now it remains to compute the w_i. This is done by setting up a linear system. Define vectors $\lambda^{ab} \in \mathbb{R}^k$ componentwise as $\lambda_i^{ab} = (u_i)_a (v_i)_b$. Define the matrix $\mathbf{W} = [w_1^\top, w_2^\top, \ldots, w_k^\top]^\top$. Now observe that

$$\underbrace{T_{abc}}_{\text{known}} = \sum_{i=1}^k \underbrace{(u_i)_a}_{\text{known}} \underbrace{(v_i)_b}_{\text{known}} \underbrace{(w_i)_c}_{\text{unknown}} = \langle \lambda^{ab}, W^{(c)} \rangle, \tag{1}$$

where $W^{(c)} = ((w_1)_c, (w_2)_c, \ldots, (w_k)_c)$ denotes the c-th column of W. This is now just some linear system, where the unknowns are the w_i.

To see that the solution to Eq. (1) is unique, we will summarize these constraints as a matrix equation. Wrap the λ^{ab} into a matrix, letting $\boldsymbol{\lambda} \in \mathbb{R}^{d_1 d_2 \times k}$ have rows which are just the λ^{ab}. Likewise, reshape T into $\mathbf{T}_{\text{matrix}} \in \mathbb{R}^{d_1 d_2 \times d_3}$; done consistently, this yields $\mathbf{T}_{\text{matrix}} = \boldsymbol{\lambda}\mathbf{W}$.

Note now that left multiplication by $\boldsymbol{\lambda}^+$ now yields $\mathbf{W}$, provided $\boldsymbol{\lambda}$ has linearly independent columns, meaning it has column rank k. Therefore, if $\boldsymbol{\lambda}$ has full column rank, then $\mathbf{W} = \boldsymbol{\lambda}^+ \mathbf{T}_{\text{matrix}}$. We conclude the proof with a lemma showing exactly this.

Lemma 2. *$\boldsymbol{\lambda}$ has full column rank.*

Proof. Assume otherwise. Note that the i-th column of $\boldsymbol{\lambda}$, denoted $\boldsymbol{\lambda}^{(i)}$, is

$$(\lambda_i^{ab})_{(a,b)\in[d_1]\times[d_2]} = ((u_i)_a (v_i)_b)_{(a,b)\in[d_1]\times[d_2]},$$

so $\boldsymbol{\lambda}^{(i)} = \operatorname{vec}(u_i \otimes v_i) = \operatorname{vec}(u_i v_i^\top)$. Then if there exists some linear dependence among the rows, we have that there exist some constants c_i, not all zero, such that

$$\begin{aligned} 0 &= \sum_{i=1}^k c_i \boldsymbol{\lambda}^{(i)} \\ &= \sum_{i=1}^k c_i u_i v_i^\top. \end{aligned}$$

Since the u_i are linearly independent, we can find some vector x which is orthogonal to $u_2, \ldots, u_k$, but not u_1. Then

$$\begin{aligned} 0 &= \sum_{i=1}^k c_i x^\top u_i v_i^\top \\ &= c_1 \langle u_1, x \rangle v_1, \end{aligned}$$

which implies $c_1 = 0$. We can repeat this for any index, yielding that $c_i = 0$ for every i, and hence no such dependence exists.

□

[1] Recall that for $\mathbf{A} \in \mathbb{R}^{n\times m}$ with linearly independent columns we have $\mathbf{A}^+\mathbf{A} = \mathbf{I}_{m\times m}$.

4 Applications

In this class, tensor decomposition and Jennrich's algorithm will be useful when applying the technique of **method of moments**. Here is the high-level idea. We are given samples from some unknown distribution $q_\theta(\cdot)$ with parameters θ. For example, in the previous lecture on Airy disks, we wanted to estimate their centers $\mu_1, \dots, \mu_k$. The idea behind method of moments is that we estimate various moments $\mathbb{E}_{x\sim q}[p(x)]$ for various polynomials p (for example p could compute the mean, variance, or anything else). This gives us a system of constraints on the parameters θ. If this system of constraints has sufficient structure, we can use tensor decomposition to efficiently solve it.

This method dates back to statistician Karl Pearson. He believed that there were some number of species of crabs, existing in different relative proportions, each of which possessed some mean characteristics, and his observations of the crabs on the island were draws from this mixture distribution. He modeled this as a classic mixture of Gaussians, and wanted to estimate the distribution over the classes, as well as the mean characteristic of each class. See also this blog post and [Moo].

4.1 Mixture of Gaussians

This section follows the approach in [HK13].

Consider the classic mixture of Gaussians setting. That is, we have unknown means $\mu_1, \dots, \mu_k \in \mathbb{R}^d$ and scaling factors $\lambda_1, \dots, \lambda_k \in [0, 1]$ satisfying $\sum_{i=1}^k \lambda_i = 1$, we are given i.i.d. samples from $q = \sum_{i=1}^k \lambda_i \mathcal{N}(\mu_i, \mathrm{Id})$, and our goal is to estimate $\{\mu_i\}, \{\lambda_i\}$ up to small errors.

To do so, we will use method of moments and Jennrich's algorithm. In this case, we will compute two moments. For our calculations below, we will view samples $x \sim q$ as first sampling $i \in [k]$ with probability λ_i, then sampling Gaussian noise $g \sim \mathcal{N}(0, \mathrm{Id})$, and finally outputting $\mu_i + g$.

First we compute $\mathbb{E}_{x\sim q}[x]$:

$$\begin{aligned}\mathbb{E}[x] &= \sum_{i=1}^k \lambda_i \mathbb{E}[\mu_i + g] \\ &= \sum_{i=1}^k \lambda_i \mu_i.\end{aligned}$$

Next we compute $\mathbb{E}_{x\sim q}[x^{\otimes 3}]$:

$$\begin{aligned}\mathbb{E}[x^{\otimes 3}] &= \sum_{i=1}^k \lambda_i \mathbb{E}\left[(\mu_i + g)^{\otimes 3}\right] \\ &= \sum_{i=1}^k \lambda_i \mathbb{E}\Big[\mu_i^{\otimes 3} + g^{\otimes 3} + \mu_i \otimes \mu_i \otimes g + \mu_i \otimes g \otimes \mu_i + g \otimes \mu_i \otimes \mu_i + \mu_i \otimes g \otimes g + g \otimes g \otimes \mu + g \otimes \mu_i \otimes g\Big].\end{aligned}$$

Luckily, many of these terms cancel. First, by symmetry of g,

$$\mathbb{E}[g^{\otimes 3}] = \mathbb{E}[\mu_i \otimes \mu_i \otimes g] = \mathbb{E}[\mu_i \otimes g \otimes \mu_i] = \mathbb{E}[g \otimes \mu_i \otimes \mu_i] = 0.$$

Now let's handle the terms with $g \otimes g$. Since g is a standard Gaussian, we have $\mathbb{E}[g \times g] = \mathrm{Id}$. Thus we can write

$$\sum_{i=1}^k \lambda_i \mathbb{E}[\mu_i \otimes g \otimes g + g \otimes g \otimes \mu_i + g \otimes \mu_i \otimes g] = \left(\sum_{i=1}^k \lambda_u \mu_i\right) \otimes_3 \mathrm{Id} = \mathbb{E}[x] \otimes_3 \mathrm{Id},$$

where we define

$$z \otimes_3 \mathrm{Id} := \sum_{a=1}^d z \otimes e_a \otimes e_a + e_a \otimes z \otimes e_a + e_a \otimes e_a \otimes z.$$

Thus, putting everything together, we have

$$\mathbb{E}[x^{\otimes 3}] = \sum_{i=1}^{k} \lambda_i \mu_i^{\otimes 3} + \mathbb{E}[x] \otimes_3 \mathrm{Id},$$

and rearranging,

$$\mathbb{E}[x^{\otimes 3}] - \mathbb{E}[x] \otimes_3 \mathrm{Id} = \sum_{i=1}^{k} \lambda_i \mu_i^{\otimes 3}.$$

We can estimate the LHS via samples. Then, we apply Jennrich's algorithm, which gives us a decomposition into components $\{v_i\}_{i=1}^k$ with $v_i := \lambda_i^{1/3} \mu_i^{\otimes 3}$. It remains to compute the λ_i. We set up a linear system to do this.

$$\begin{aligned}\mathbb{E}[x] &= \sum_{i=1}^{k} \lambda_i \mu_i \\ &= \sum_{i=1}^{k} \lambda_i^{2/3} v_i\end{aligned}$$

If the μ_i are all linearly independent, then because $k \leq d$ (needed to even run Jennrich's), this system has a unique solution.

4.2 Mixture of Exponentials

This section follows the approach in [HK15].

Recall in last lecture, we learned how to use the matrix pencil method to recover the centers $\mu_1, \ldots, \mu_k$ of Airy disks, assuming these centers lie in $\mathbb{R}$. Let's use Jennrich's algorithm to generalize this to centers in $\mathbb{R}^2$.

Fix centers $\mu_1, \ldots, \mu_k \in \mathbb{R}^2$. Recall we are given access to the Fourier transform

$$G : \omega \to \frac{1}{k} \sum_{j=1}^{k} e^{2\pi i \langle \omega, \mu_j \rangle}$$

for any $\omega \in \mathbb{R}^2$ with $||\omega|| \leq 1$. Our goal is to recover $\mu_1, \ldots, \mu_k$. We can do by applying Jennrich's algorithm as follows:

1. Fix some $m \in \mathbb{N}$ to be sufficiently large.
2. Sample $\omega_1, ..., \omega_m \in \mathbb{R}^2 \sim B(0.49)$ and $v, v' \sim \mathbb{S}^1$. Here, $B(0.49)$ denotes the 2D ball of radius 0.49 around the origin.
3. Define $x_1 := 0.02 \cdot v$ and $x_2 := 0.02 \cdot v'$.
4. Define $T \in \mathbb{R}^{m \times m \times 2}$ where $T_{abc} := G(\omega_a + \omega_b + x_c)$ for all $a, b \in [m]$ and $c \in [2]$. (Note: the constants 0.49 and 0.02 from the previous steps are chosen to ensure that the input $\omega_a + \omega_b + c$ into G has norm ≤ 1)
5. Define $u_i \in \mathbb{R}^d$ and $w_i \in \mathbb{R}^2$ such that $(u_i)_a := e^{2\pi i \langle w_a, \mu_j \rangle}$ and $(w_i)_c := e^{2\pi i \langle x_c, \mu_j \rangle}$ for $a \in [d]$ and $c \in [2]$. Then we can write $T_{abc} = \frac{1}{k} \sum_{j=1}^{k} (u_i)_a (u_i)_b (w_i)_c$, so $T = \frac{1}{k} \sum_{j=1}^{k} u_i \otimes u_i \otimes w_i$.
6. As long as $\{u_i\}$ are linearly independent and $\{w_i\}$ are not collinear, we can apply Jennrich's algorithm to obtain the $\{u_i\}$ and $\{w_i\}$, which in turn can be used to solve for the centers $\{\mu_j\}$.

How does this compare to the matrix pencil method? Last time, we applied G to vectors in a evenly spaced grid, which gave us structured data, that we formed into two (Hankel) matrices UU^T and UDU^T, and we solved for the centers by taking an eigendecomposition. This time, we are applying G to (carefully chosen) random vectors, forming matrices UD_ZU^T and $UD_{Z'}U^T$, and again solving for the centers by taking an eigendecomposition. Thus, Jennrich's algorithm can be viewed as a generalization of the matrix pencil method.

References

[HK13] Daniel Hsu and Sham M. Kakade. Learning mixtures of spherical gaussians: Moment methods and spectral decompositions. In *Proceedings of the 4th Conference on Innovations in Theoretical Computer Science*, ITCS '13, page 11–20, New York, NY, USA, 2013. Association for Computing Machinery.

[HK15] Qingqing Huang and Sham M. Kakade. Super-resolution off the grid. *CoRR*, abs/1509.07943, 2015.

[HL09] Christopher J. Hillar and Lek-Heng Lim. Most tensor problems are NP hard. *CoRR*, abs/0911.1393, 2009.

[LRA93] S. E. Leurgans, R. T. Ross, and R. B. Abel. A decomposition for three-way arrays. *SIAM Journal on Matrix Analysis and Applications*, 14(4):1064–1083, 1993.

[Mat] Will the real jennrich's algorithm please stand up? `https://www.mathsci.ai/post/jennrich/`. Accessed: 2023-09-14.

[Moo] Pearson's polynomial. `http://blog.mrtz.org/2014/04/22/pearsons-polynomial.html`. Accessed: 2023-09-14.