CMS 165 Foundations of Machine Learning Homework 3 Solutions

An orthogonal decomposition of a symmetric tensor $T \in \mathbb{R}^{n\times n\times n}$ is a collection of orthonormal vectors $\{v_1, v_2, \ldots, v_k\}$ together with corresponding positive scalars $\lambda_i > 0$ such that

$$T = \sum_{j=1}^{k} \lambda_j v_j^{\otimes 3} \tag{1}$$

In this question we will consider the variational characterization of this decomposition. First, consider the variational characterization of the eigenvalues of a symmetric matrix M. Let λ_i denote the eigenvalues and assume $\lambda_1 > \lambda_2 > \ldots \lambda_k$. Then the *Rayleigh quotient*

$$\frac{u^T M u}{u^T u} \tag{2}$$

is maximized over non-zero vectors by the eigenvector associated with λ_1. Recalling the multilinear tensor notation from class, one can write the mapping as follows,

$$u \to \frac{u^T M u}{u^T u} \equiv u \to \frac{M(u,u)}{u^T u} \tag{3}$$

In tensors, the notion of eigenvectors becomes a little different than matrix case. For an extensive discussion please refer to Section 4 of [1]. For a third order tensor T, the orthogonal decomposition given in (1) is unique and $\{v_1, v_2, \ldots, v_k\}$ are called robust eigenvectors. Consider the *generalized Rayleigh quotient*

$$u \to \frac{T(u,u,u)}{(u^T u)^{3/2}} \tag{4}$$

and consider the following optimization problem

$$\max_{u\in\mathbb{R}^n} T(u,u,u) \;\; s.t. \;\; \|u\| \leq 1. \tag{5}$$

Question 1. *True/False* The robust eigenvectors $\{v_1, v_2, \ldots, v_k\}$ are the only stationary points of this maximization. Explain your answer in detail and compare it with matrix case.
Hint: [1] is a good resource for solving this question. [1]

Solution: **FALSE**. Take the set of robust eigenvectors $\{v_1, \ldots, v_k\}$ to be orthonormal and associate corresponding eigenvalues $\lambda_1, \ldots, \lambda_k \in \mathbb{R}$. We have the following optimization problem:

$$\begin{aligned} \max_{u\in\mathbb{R}^n} \quad & T(u,u,u) \\ \text{s.t.} \quad & ||u|| \leq 1 \end{aligned} \tag{6}$$

Forming the Lagrangian

$$\begin{aligned} \mathcal{L}(u,\lambda) &= T(u,u,u) - \frac{3}{2}\lambda(||u|| - 1) \\ &= \sum_{i=1}^{k} \lambda_i (v_i^\top u)^3 v_i - \frac{3}{2}\lambda u^\top u + \frac{3}{2}\lambda \end{aligned}$$

[1] We thank Lucien Werner for his contributions to the solutions

We get the stationary point equation by differentiating $\mathcal{L}(u, \lambda)$ and setting the result equal to 0.

$$\begin{aligned}
\nabla_u \mathcal{L}(u, \lambda) &= 3 \sum_{i=1}^{k} \lambda_i (v_i^\top u)^2 v_i - 3\lambda u \\
&= 3T(I, u, u) - 3\lambda u \\
&= 0 \\
\Rightarrow\ T(I, u, u) &= \lambda u \qquad (7)
\end{aligned}$$

Any vector u that satisfies (7) is a stationary point of the maximization (6). We define a particular u as follows:

$$u := \lambda \sum_{i=1}^{k} \frac{v_i}{\lambda_i}$$

Note that

$$v_i^\top u = v_i^\top \lambda \left(\frac{v_1}{\lambda_1} + \ldots + \frac{v_k}{\lambda_k} \right) = \lambda \frac{v_i^\top v_i}{\lambda_i} = \lambda \frac{1}{\lambda_i} \qquad \text{(by orthogonality)}$$

Plugging u into (7)

$$\begin{aligned}
T(I, u, u) &= \sum_{i=1}^{k} \lambda_i (v_i^\top u)^2 v_i && (8) \\
&= \sum_{i=1}^{k} \lambda_i \left(\lambda \frac{1}{\lambda_i} \right)^2 v_i && (9) \\
&= (\lambda)^2 \sum_{i=1}^{k} \frac{v_i}{\lambda_i} && (10) \\
&= \lambda u && (11)
\end{aligned}$$

This demonstrates that a stationary point of the maximization (6) does not have to be one of the v_is. Therefore the claim is FALSE.

Now consider the matrix case. Keeping the tensor notation, we have

$$\begin{aligned}
M(u, u) &:= \sum_{i=1}^{k} \lambda_i (v_i^\top u)^2 && (12) \\
M(u, I) &:= \sum_{i=1}^{k} \lambda_i (v_i^\top u) v_i && (13)
\end{aligned}$$

Then, forming the optimization (6) with $M(u, u)$ instead of $T(u, u, u)$ and forming the Lagrangian

$$\mathcal{L}(u, \lambda) = M(u, u) - \lambda(||u|| - 1)$$

Taking the derivative and setting it equal to 0,

$$\nabla_u \mathcal{L}(u, \lambda) = 2M(u, I) - 2\lambda u \tag{14}$$
$$= 0 \tag{15}$$
$$\Rightarrow \; M(u, I) = \lambda u \tag{16}$$

Let u be defined as follows for the moment:

$$u = \sum_{i=1}^{k} \gamma_i v_i$$

Then

$$M(u, I) = \lambda u \tag{17}$$
$$\Leftrightarrow \; \sum_{i=1}^{k} \lambda_i (v_i^\top u) v_i = \lambda u \tag{18}$$
$$\Leftrightarrow \; \sum_{i=1}^{k} \lambda_i \gamma_i v_i = \lambda \sum_{i=1}^{k} \gamma_i v_i \tag{19}$$

This means

$$\sum_{i=1}^{k} \lambda_i \gamma_i = \lambda \sum_{i=1}^{k} \gamma_i \tag{20}$$

If we take $\lambda_1 > \lambda_2 > \cdots > \lambda_k$ then (20) implies that all of the $\gamma_i = 0$ except for a single one at a time. Then by our construction of u, a stationary point of the maximization can only be $v_1, \ldots, v_k$. From this we can also see that we should pick $u = v_1$ to maximize $M(u, u)$. This contrasts with the tensor case above where we could choose u to be a linear combination of the robust eigenvectors with all of the coefficients being non-zero.

In [1], the convergence analysis of tensor power method is shown for a tensor T which has an orthogonal tensor decomposition. In this question, we will look at the case where tensors have non-orthogonal tensor decomposition. Non-orthogonal tensor decompositions can be reduced to orthogonal tensor decomposition and then tensor power method can be used to recover rank-1 components of orthogonal tensor decomposition.

Question 2. Mathematically describe the conversion of non-orthogonal tensor decomposition to orthogonal decomposition. How can we obtain the non-orthogonal components from the output of tensor power method?

Hint: In order to answer this question and for further details, it would be good to refer to TensorBook on Piazza.

Solution: First, we perform the whitening procedure. Let $T \in \mathbb{R}^{d \times d \times d}$ be a symmetric tensor such that

$$T = \sum_{j=1}^{k} \lambda_j \cdot a_j^{\otimes 3}$$

where a_j's need not be orthogonal but wlog $||a_j|| = 1$. Assume we have a corresponding symmetric matrix with the same rank-1 components:

$$M = \sum_{j=1}^{k} \tilde{\lambda}_j \cdot a_j^{\otimes 2}$$

If we don't already have such a matrix, we can obtain one through a random combination of the 2D slices of T; $M := T(I, I, \theta) \in \mathbb{R}^{d \times d}$ with $\theta \sim \mathbb{N}(0, I_d)$. Taking eigendecomposition of M, we obtain

$$M = U \text{ diag}(\gamma) U^T$$

with $U \in \mathbb{R}^{d \times k}$ and $\gamma \in \mathbb{R}^k$. Let $W \in \mathbb{R}^{d \times k}$ denote the whitening matrix such that

$$W := U \text{ diag} \left(|\gamma|^{-1/2} \right)$$

Now, plugging this W into T

$$\begin{aligned} T(W, W, W) &= \sum_{j=1}^{k} \lambda_j (W^\top a_j)^{\otimes 3} \\ &= \sum_{j=1}^{k} \underbrace{\frac{\lambda_j}{\tilde{\lambda}_j^{3/2}}}_{:=\mu_j} \underbrace{\left(W^\top a_j \sqrt{\tilde{\lambda}_j} \right)^{\otimes 3}}_{:=v_j} \end{aligned}$$

Next we construct

$$V := W^\top A \text{ diag} \left(\tilde{\lambda}^{1/2} \right) = \left[W^\top a_1 \sqrt{\tilde{\lambda}_1} \;\middle|\; \cdots \;\middle|\; W^\top a_k \sqrt{\tilde{\lambda}_k} \right] \in \mathbb{R}^{k \times k}$$

With this V observe

$$VV^\top = W^\top A \text{ diag}(\tilde{\lambda}) A^\top W = W^\top M W = I_k = V^\top V$$

Therefore the v_j's that comprise V form an orthonormal basis for the tensor $T(W, W, W)$. To determine what these v_j's are, we can apply the tensor power method, a process that we explored in more detail in Problem 1. With the v_j's in hand, we then reverse the whitening process to get the non-orthogonal rank-1 components, i.e., the a_j's.

Recall the definitions from the whitening procedure; specifically $V = W^\top A \text{ diag}\left(\tilde{\lambda}^{1/2}\right)$, $M = U \text{ diag}(\gamma) U^T$, and $W = U \text{ diag}\left(|\gamma|^{-1/2}\right)$. Then

$$U \text{ diag}\left(\gamma^{1/2}\right) V = U \text{ diag}\left(\gamma^{1/2}\right) \text{ diag}\left(|\gamma|^{-1/2}\right) U^\top A \text{ diag}\left(\tilde{\lambda}^{1/2}\right) = UU^\top A \text{ diag}\left(\tilde{\lambda}^{1/2}\right) \quad (21)$$

Since $UU^\top A = A$ then from (21) we have

$$\begin{aligned} U \text{ diag}\left(\gamma^{1/2}\right) V &= A \text{ diag}\left(\tilde{\lambda}^{1/2}\right) \\ \Rightarrow\ A &= U \text{ diag}\left(\gamma^{1/2}\right) V \text{ diag}\left(\tilde{\lambda}^{-1/2}\right) \end{aligned}$$

This is equivalent to saying

$$a_j = \frac{1}{\sqrt{\tilde{\lambda}_j}} U \text{ diag}\left(\gamma^{1/2}\right) v_j, \qquad j = 1, \ldots, k$$

We see that it is possible to recover the non-orthogonal rank-1 components a_j from the rank-1 orthogonal components v_j—which themselves have been computed using the tensor power method—using the unwhitening procedure described here.

Question 3. A very useful trick in non-convex optimization problems is to find hidden convexity in the problem and solve the non-convex problem by converting it to convex form. Examples can be found in [2] and [3]. Specifically in [2], Belkin et al. discussed that many important problems in machine learning can be interpreted as basis learning and demonstrated that they can be formulated as gradient ascent. Following the discussion in [2], show how the matrix eigenvector recovery and tensor orthogonal decomposition problems can be reformulated.

Solution: Following the terminology in [2], we define the basis encoding function for the matrix eigenvector recovery problem as

$$F(u) := u^\top Au$$

for a symmetric matrix A. The gradient iteration in the Belkin algorithm is

$$u \longleftarrow \frac{\nabla_u F(u)}{||\nabla_u F(u)||}$$

For $F(u) := u^\top Au$ we get

$$u \longleftarrow \frac{Au}{||Au||}$$

which is the definition of the power iteration for the matrix A. If we assume the eigenvalues of A satisfy $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_k| > 0$ then repeated iterations of the algorithm will converge to a vector u that is associated with largest eigenvalue λ_1. To see this, let u_0 be a non-zero vector and let $v_1, \ldots, v_k$ be the eigenvectors of A (they form a basis since A is real symmetric and therefore diagonalizable by the Spectral Theorem). We can write

$$u_0 = \sum_{i=1}^{k} c_i v_i$$

for some $c_i \in \mathbb{R}$. Then the power iterations are

$$(1) \quad u \longleftarrow \frac{Au_0}{||Au_0||} = \frac{A\sum_{i=1}^k c_i v_i}{||A\sum_{i=1}^k c_i v_i||} = \frac{\sum_{i=1}^k \lambda_i c_i v_i}{||\sum_{i=1}^k \lambda_i c_i v_i||} \tag{22}$$

$$(2) \quad u \longleftarrow \frac{A^2u_0}{||A^2u_0||} = \frac{A\sum_{i=1}^k \lambda_i c_i v_i}{||A\sum_{i=1}^k \lambda_i c_i v_i||} = \frac{\sum_{i=1}^k \lambda_i^2 c_i v_i}{||\sum_{i=1}^k \lambda_i^2 c_i v_i||} \tag{23}$$

$$\vdots \tag{24}$$

$$(N) \quad u \longleftarrow \frac{A^Nu_0}{||A^Nu_0||} = \frac{A\sum_{i=1}^k \lambda_i^{N-1} c_i v_i}{||A\sum_{i=1}^k \lambda_i^{N-1} c_i v_i||} = \frac{\sum_{i=1}^k \lambda_i^N c_i v_i}{||\sum_{i=1}^k \lambda_i^N c_i v_i||} \tag{25}$$

$$\tag{26}$$

Taking the last expression and factoring

$$\frac{\sum_{i=1}^k \lambda_i^N c_i v_i}{||\sum_{i=1}^k \lambda_i^N c_i v_i||} = \frac{c_1\lambda_1^N\left(v_1 + \sum_{i=2}^k \frac{c_i}{c_1}\left(\frac{\lambda_i}{\lambda_1}\right)^N v_i\right)}{\left|\left|c_1\lambda_1^N\left(v_1 + \sum_{i=2}^k \frac{c_i}{c_1}\left(\frac{\lambda_i}{\lambda_1}\right)^N v_i\right)\right|\right|}$$

As N becomes large the term $\left(\frac{\lambda_i}{\lambda_1}\right)^N \longrightarrow 0$. (The speed at which is converges is determined by the ratio λ_1/λ_2.) So we have after N iterations of the power method that

$$u \approx \frac{c_1\lambda_1^N\,(v_1+0)}{||c_1\lambda_1^N\,(v_1+0)\,||} = \frac{c_1\lambda_1^N v_1}{c_1\lambda_1^N||v_1||} = \frac{v_1}{||v_1||}$$

for $c_1 \neq 0$. Thus u is just the normalized dominant eigenvector of A. To compute the remaining eigenvectors, we can "remove" the first eigendimension from A as follows:

$$A' = A - \lambda_1 \frac{v_1 v_1^\top}{||v_1||^2}$$

Then re-run the power method on A', A'' etc. to recover v_2, v_3 etc. At the end of the process we will have recovered all of the eigenvectors of A.

For orthogonal tensor decomposition (again following the Belkin notation), we use the tensor power method to recover the orthogonal decomposition of a tensor

$$T = \sum_{k=1}^{m} w_k a_k^{\otimes r}$$

where $w_k \in \mathbb{R}\{0\}$ and $a_k \in \mathbb{R}^d$. The tensor power method converges to the robust eigenvectors $v_1, \ldots, v_m$ of T when the a_k are orthogonal (we saw this in Problem 1). Define the BEF as

$$F(u) := Tu^r = \sum_{k=1}^{m} w_k (u^\top a_k)^r$$

Then the Belkin algorithm gradient iteration is

$$u \longleftarrow \frac{\nabla_u F(u)}{||\nabla_u F(u)||} \tag{27}$$

$$= \frac{\nabla_u T u^r}{||\nabla_u T u^r||} \tag{28}$$

$$= \frac{\nabla_u \sum_{k=1}^m w_k (u^\top a_k)^r}{||\nabla_u \sum_{k=1}^m w_k (u^\top a_k)^r||} \tag{29}$$

$$= \frac{r\sum_{k=1}^m w_k (u^\top a_k)^{r-1} a_k}{r||\sum_{k=1}^m w_k (u^\top a_k)^{r-1} a_k||} \tag{30}$$

$$= \frac{\sum_{k=1}^m w_k (u^\top a_k)^{r-1} a_k}{||\sum_{k=1}^m w_k (u^\top a_k)^{r-1} a_k||} \tag{31}$$

$$= \frac{Tu^{r-1}}{||Tu^{r-1}||} \tag{32}$$

This last expression is in the form of the power iteration for orthogonal tensor decomposition. Thus, both this and the matrix eigenvector problems can be formulated as finding the solution to a fixed point iteration of the gradient of a basis encoding function.

Question 4. You can refer to [5] to answer this problem.

4.1. Briefly explain what is a graphical model, in terms of the conditional independencies among the random variables in the model.

4.2. What is a tensor network diagram?

4.3. Refer to definitions 1.1 to 1.4 in the [5]. Explain why a graphical model associated to a graph $H = (U, \mathcal{C})$ with clique ψ_C is the same as the data of a tensor network associated to its dual graph $H^\star$ with tensors ψ_c at each vertex of $H^\star$.

Solution:

4.1

We define a graphical model as

> a graph where the nodes correspond to random variables in the model, the edges correspond to the presence/absence of conditional dependencies between variables, and the edge weights correspond to the strength of the conditional dependencies between variables.

The main classes of graphical models are Bayesian networks and Markov random fields (MRF). The former are DAGs and the latter are undirected graphs which may have cycles. Bayesian networks have nice properties that allow us to represent the joint probability distribution of the variables in the model in a factored form. Mathematically, this is

$$P(X_1, \ldots, X_n) = \prod_{i=1}^{n} p(X_i | X_{j \to i})$$

where $p(X_i|X_{j\to i})$ is the probability distribution of X_i conditioned on all other variables that are parents of X_i. A Markov random field does not have the nice factorization in general but it does satisfy Markov properties, namely that every variable X_i is independent from all other variables conditioned on its immediate neighbours. In particular cases, a MRF can be factorized; for example into a decomposition over its cliques:

$$P(X) = \prod_{C \in \text{cliques}(G)} \psi_C(X_C)$$

where $\psi_C(X_C)$ is the probability distribution of clique C.

4.2

Tensor networks are a tool to visualize tensor operations. To illustrate how they work, we will present a particularly important operation: contraction over the indices. Here's an example. Take two rank-2 tensors (i.e., matrices) A and B of appropriate dimension so that the matrix product AB makes sense. Then a contraction C over the j index gives AB:

$$AB = C_{ik} = \sum_j A_{ij} C_{jk} \tag{33}$$

Notice that C_{ik} is a matrix, so an index contraction does not necessarily result in a scalar. A case where the contraction is a scalar is the scalar product of two rank-1 tensors (i.e., vectors) $a, b \in \mathbb{R}^n$:

$$a^\top b = C = \sum_{i=1}^{n} a_i b_i \tag{34}$$

A more complicated example is the contraction of two rank-4 tensors $A, C \in \mathbb{R}^{n\times n\times n\times n}$ and two rank-3 tensors $B, D \in \mathbb{R}^{n\times n\times n}$. The following contraction results in a scalar:

$$E = \sum_{i,j,k,l,m,n,p=1}^{n} A_{ijlk} B_{jkp} C_{lnpm} D_{nmi} \in \mathbb{R} \tag{35}$$

To represent tensors graphically consider the pictorial notation in Figure 1.

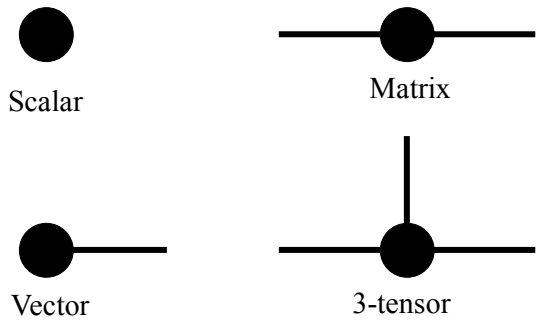
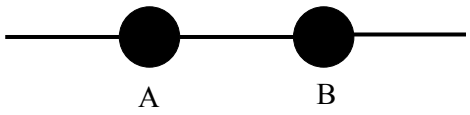
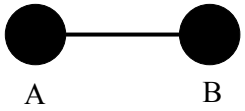


Figure 1: Tensor graphical notation

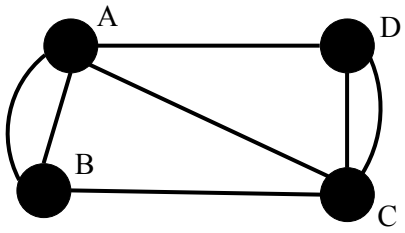
Using this notation, we can represent the contractions over indices as networks, where the nodes are tensors and the number of arms indicates the order of the tensor and over which dimension of its neighbouring tensors it is being contracted. In particular, the example in (33) can be represented as the following tensor network diagram:



The scalar product in (34) can be represented as:



Finally, the complicated tensor contraction in (35) can be represented as:



The most illustrative example is that of the trace operation, which can be represented as a cyclic graph. This concisely demonstrates the cyclic properties of the trace.

N.B. Some of the examples in this section were presented from [4].

4.3

A tensor network is a representation of a joint probability distribution that can be factored based on the conditional dependency properties in the graphical model generating the distribution. Let the tensor P be associated with hypergraph $H = (U, C)$. Considering the incidence matrix M of H, we interpret each of its rows as being one of the vertex supersets $u \in U$ and each of its columns being one of the cliques $c \in C$. (This is analogous to the incidence matrix of a normal graph forming a mapping between vertices and edges). Then P can be represented by a network of the potential functions of the cliques

$$\psi_C : \prod_{u \in C} \mathcal{X}_u \longrightarrow C$$

which are equivalently represented as tensors $v_j^{\otimes |\mathcal{X}_u|,\ u \in C}$.

The dual graph of H, H^*, has incidence matrix $M^\top$ by definition of the dual graph (edges become nodes, nodes become edges, so we just transpose the node-to-edge mapping). We then associate tensors

$$T_c = \psi_c \in \bigotimes_{v_c \in e_u} \mathbb{R}^{|\mathcal{X}_u|}$$

to each node of H^* (equivalently each edge of H). Thus we have defined a tensor network on H^*. Finally, the probability distribution P associated with the tensor network of H can be written (up to

normalization) as

$$P(x_u \text{ for } u \in U) = \prod_{c \in C} \psi_c(X_c) = \prod_{c \in C} (T_c)_{X_c}$$

In other words, the probability distribution P associated with H, which has cliques described by potential functions $\phi_{c,\, c \in C}$, is equivalent to one corresponding to a graph H^* which, by construction, has tensors T_c for each $\phi_{c,\, c \in C}$ at its vertices.

References

[1] Animashree Anandkumar, Rong Ge, Daniel Hsu, Sham M Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models. *The Journal of Machine Learning Research*, 15(1):2773–2832, 2014.

[2] Mikhail Belkin, Luis Rademacher, and James Voss. Basis learning as an algorithmic primitive. In *Conference on Learning Theory*, pages 446–487, 2016.

[3] Yurii Nesterov and Boris T Polyak. Cubic regularization of newton method and its global performance. *Mathematical Programming*, 108(1):177–205, 2006.

[4] Román Orús. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. *Annals of Physics*, 349:117–158, 2014.

[5] Elina Robeva and Anna Seigal. Duality of graphical models and tensor networks. *Information and Inference: A Journal of the IMA*, 2017.