Probabilistic Latent Tensor Factorization

Y. Kenan Yılmaz and A. Taylan Cemgil**

Department of Computer Engineering,
Boğaziçi University, 34342 Bebek, Istanbul, Turkey
kenan@sibnet.com.tr, taylan.cemgil@boun.edu.tr

Abstract. We develop a probabilistic framework for multiway analysis of high dimensional datasets. By exploiting a link between graphical models and tensor factorization models we can realize any arbitrary tensor factorization structure, and many popular models such as CP or TUCKER models with Euclidean error and their non-negative variants with KL error appear as special cases. Due to the duality between exponential families and Bregman divergences, we can cast the problem as inference in a model with Gaussian or Poisson components, where tensor factorisation reduces to a parameter estimation problem. We derive the generic form of update equations for multiplicative and alternating least squares. We also propose a straightforward matricisation procedure to convert element-wise equations into the matrix forms to ease implementation and parallelisation.

Key words: Tensor factorisation, Non-negative decompositions, NMF, NTF, CP, TUCKER, EM algorithm, Graphical models

1 Introduction

Advances in computing power, data acquisition, storage technologies made it possible to collect and process huge amounts of data in many disciplines. Yet, in order to extract useful information effective and efficient computational tools are needed. In this context, matrix factorisation techniques have emerged as a useful paradigm [10,15]. Clustering, ICA, NMF, LSI, collaborative filtering and many such methods can be expressed and understood as matrix factorisation problems. Thinking of a matrix as the basic data structure maps well onto special purpose hardware (such as a GPU unit) to make algorithms run faster via parallelisation. Moreover, matrix computations come with a toolbox of well understood algorithms with precise error analysis, performance guarantees and extremely efficient standard implementations (e.g., SVD).

A useful method in multiway analysis is *tensor factorization* (TF) to extract hidden structure in data that consists of more than two entities. However, since there are many more natural ways to factorise a multiway array, there exists a plethora of related models with distinct names discussed in detail in recent

** This research is funded by the Bogazici University Research Fund under grant BAP 09A105P.

tutorial reviews [9,1]. A recent book [5] outlined various optimization algorithms for non-negative TF for alpha and beta divergences. The idea of sparse non-negative TUCKER was discussed in [12]. Use of the probabilistic approach, then, for the matrix factorization (PMF) was presented by [13] while probabilistic non-negative TF came out in [14]. However, all these works focus on isolated models.

The motivation behind this paper is pave the way to a unifying framework in which any arbitrary TF structure can be realized and the associated inference algorithm can be derived automatically using matrix computation primitives. For this, we introduce a notation for TF models that closely resembles probabilistic graphical models [7]. We also propose a probabilistic approach to multiway analysis as this provides a natural framework for model selection and handling missing values. We focus on using the KL divergence and the Euclidean metric to cover both unconstrained and non-negative decompositions. Our probabilistic treatment generalises the statistical treatment of NMF models described in [4,6].

2 Tensor Factorization (TF) Model

Following the established jargon, we call a N-way array $X \in \mathcal{X}^{I_1 \times I_2 \times \cdots \times I_N}$ simply a 'tensor'. Here, I_n are finite index sets, where i_n is the corresponding index. We denote an element of the tensor $X(i_1, i_2, \ldots, i_N) \in \mathcal{X}$ as $X^{i_1,i_2,\ldots,i_N}$. Similarly, given the index set $W = \{i_1, \ldots, i_N\}$ we use the notation $X(w)$ to denote an element of $X^{i_1,i_2,\ldots,i_N}$.

We associate with each TF model an undirected graph, where each vertex corresponds to an index. We let V be the set of vertices $V = \{v_1, \ldots v_n, \ldots, v_N\}$. Our objective is to estimate a set of tensors $\mathcal{Z} = \{Z_\alpha | \alpha = 1 \ldots N\}$ such that

$$\text{minimize } d(X||\hat{X}) \text{ s.t. } \hat{X}(w) = \sum_{\bar{w} \in \bar{W}} \prod_{\alpha} Z_\alpha(v_\alpha) \tag{1}$$

where the function d is a suitable error measure, which we define later. Each Z_α is associated with an index set V_α such that $V = \cup_\alpha V_\alpha$. Two distinct sets V_α and $V_{\alpha'}$ can have nonempty intersection but they don't contain each other. We define a set of 'visible' indices $W \subseteq V$ and 'invisible' indices $\bar{W} \subseteq V$ such that $W \cup \bar{W} = V$ and $W \cap \bar{W} = \emptyset$.

Example 1 (TUCKER3 Factorization). The TUCKER3 factorization [8,9] aims to find Z_α for $\alpha = 1 \ldots 4$ that solve the following optimization problem where in our notation, the TUCKER3 model is given by $N = 4$, $V = \{p, q, r, i, j, k\}$, $V_1 = \{i, p\}$, $V_2 = \{j, q\}$, $V_3 = \{k, r\}$, $V_4 = \{p, q, r\}$ and $W = \{i, j, k\}$, $\bar{W} = \{p, q, r\}$.

$$\text{minimize } d(X||\hat{X}) \text{ s.t. } \hat{X}^{i,j,k} = \sum_{p,q,r} Z_1^{i,p} Z_2^{j,q} Z_3^{k,r} Z_4^{p,q,r} \qquad \forall i, j, k \tag{2}$$

In this paper for the error measure d, we use KL divergence and Euclidean distance that give rise to two variants that we call as PLTF$_{KL}$ (*Probabilistic Latent Tensor Factorization*) and PLTF$_{EU}$ respectively. Alternatives such as NMF

with IS divergence also exist in [6]. Due to the duality between the Poisson likelihood and KL divergence, and between the Gaussian likelihood and Euclidean distance [3], solving the TF problem in (1) is equivalent to finding the ML solution of $p(X|Z_{1:N})$ [6].

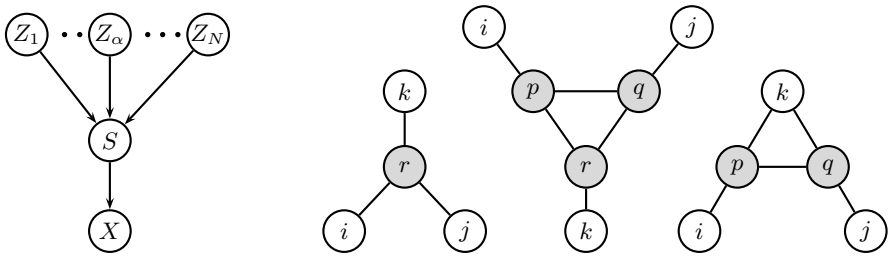

Fig. 1. The DAG on the left is the graphical model of PLTF. X is the observed multiway data and Z_α's are the parameters. The latent tensor S allows us to treat the problem in a data augmentation setting and apply the EM algorithm. On the other hand, the factorisation implied by TF models can be visualised using the semantics of undirected graphical models where cliques (fully connected subgraphs) correspond to individual factors. The undirected graphs on the right represent CP, TUCKER3 and PARATUCK2 models in the order. The shaded indices are hidden, i.e., correspond to the dimensions that are not part of X.

2.1 Probability Model

For $PLTF$, we write the following generative model such that $W \cup \bar{W} = \cup_\alpha V_\alpha = V$ and for their instantiations $(w, \bar{w}) = \cup_\alpha v_\alpha = v$

$$\Lambda(v) = \prod_\alpha^N Z_\alpha(v_\alpha) \qquad \text{model paramaters to estimate} \tag{3}$$

$$S(w,\bar{w}) \sim \mathcal{PO}(S; \Lambda(v)) \qquad \text{element of latent tensor for } PLTF_{KL} \tag{4}$$

$$S(w,\bar{w}) \sim \mathcal{N}(S; \Lambda(v), 1) \qquad \text{element of latent tensor for } PLTF_{EU} \tag{5}$$

$$X(w) = \sum_{\bar{w}\in\bar{W}} S(w,\bar{w}) \qquad \text{model estimate after augmentation} \tag{6}$$

$$M(w) = \begin{cases} 0 & X(w) \text{ is missing} \\ 1 & \text{otherwise} \end{cases} \qquad \text{mask array} \tag{7}$$

Note that due to reproductivity property of Possion and Gaussian distributions [11] the observation $X(w)$ has the same type of distribution as $S(w, \bar{w})$.

Next, $PLTF$ handles the missing data smoothly by the following observation model [13,4]

$$p(X|S)p(S|Z_{1:N}) = \prod_{w\in W}\prod_{\bar{w}\in\bar{W}} \left(p(X(w)|S(w,\bar{w}))\; p(S(w,\bar{w})|Z_{1:N})\right)^{M(w)} \tag{8}$$

2.2 PLTF$_{KL}$ Fixed Point Update Equation

We can easily optimise for Z_α by an EM algorithm. The loglikelihood $\mathcal{L}_{KL}$ is

$$\sum_{w\in W}\sum_{\bar{w}\in\bar{W}} M(w)\left(S(w,\bar{w})\log\left(\prod_\alpha Z_\alpha(v_\alpha)\right)-\left(\prod_\alpha Z_\alpha(v_\alpha)\right)-\log S(w,\bar{w})!\right)$$

subject to the constraint $X(w)=\sum_{\bar{w}} S(w,\bar{w})$ whenever $M(w)=1$. The E-step is calculated by identifying the posterior of S as a multinomial distribution [11] with the following sufficient statistics

$$\langle S(w,\bar{w})\rangle = \frac{X(w)\prod_\alpha Z_\alpha(v_\alpha)}{\sum_{\bar{w}\in\bar{W}}\prod_\alpha Z_\alpha(v_\alpha)} = \frac{X(w)}{\hat{X}(w)}\prod_\alpha Z_\alpha(v_\alpha) \tag{9}$$

where $\hat{X}(w)$ is the model estimate as $\hat{X}(w)=\sum_{\bar{w}\in\bar{W}}\prod_\alpha Z_\alpha(v_\alpha)$. The M-step is

$$\frac{\partial\mathcal{L}_{KL}}{\partial Z_\alpha(v_\alpha)}=0 \qquad \Rightarrow \qquad Z_\alpha(v_\alpha)=\frac{\sum_{v\notin V_\alpha} M(w)\,\langle S(w,\bar{w})\rangle}{\sum_{v\notin V_\alpha} M(w)\prod_{\alpha'\neq\alpha} Z_{\alpha'}(v_{\alpha'})} \tag{10}$$

After substituting (9) in (10) we obtain the following fixed point update for Z_α

$$Z_\alpha(v_\alpha)\leftarrow Z_\alpha(v_\alpha)\frac{\sum_{v\notin V_\alpha} M(w)\frac{X(w)}{\hat{X}(w)}\prod_{\alpha'\neq\alpha} Z_{\alpha'}(v_{\alpha'})}{\sum_{v\notin V_\alpha} M(w)\prod_{\alpha'\neq\alpha} Z_{\alpha'}(v_{\alpha'})} \tag{11}$$

Definition 1. *We define the tensor valued function $\Delta_\alpha(A):\mathbb{R}^{|A|}\to\mathbb{R}^{|Z_\alpha|}$ (associated with Z_α) as*

$$\Delta_\alpha(A)\equiv\left[\sum_{v\notin V_\alpha}\left(A(w)\prod_{\alpha'\neq\alpha} Z_{\alpha'}(v_{\alpha'})\right)\right] \tag{12}$$

$\Delta_\alpha(A)$ is an object the same size of Z_α. We also use the notation $\Delta_{Z_\alpha}(A)$ especially when Z_α are assigned distinct letters. $\Delta_\alpha(A)(v_\alpha)$ refers to a particular element of $\Delta_\alpha(A)$. Using this new definition, we rewrite (11) more compactly as

$$Z_\alpha\leftarrow Z_\alpha\circ\Delta_\alpha(M\circ X/\hat{X})/\Delta_\alpha(M) \tag{13}$$

where $\circ$ and / stand for element wise multiplication (Hadamard product) and division respectively. Later we develop the explicit matrix forms of these updates.

2.3 PLTF$_{EU}$ Fixed Point Update Equation

The derivation follows closely Section 2.2 where we merely replace the Poisson likelihood with that of a Gauissian. The complete data loglikelihood becomes

$$\mathcal{L}_{EU}=\sum_{w\in W}\sum_{\bar{w}\in\bar{W}} M(w)\left(-\frac{1}{2}\log(2\pi)-\frac{1}{2}\left(S(w,\bar{w})-\prod_\alpha Z_\alpha(v_\alpha)\right)^2\right) \tag{14}$$

subject to the constraint $X(w) = \sum_{\bar{w}} S(w, \bar{w})$ for $M(w) = 1$. The sufficient statistics of the Gaussian posterior $p(S|Z, X)$ are available in closed form as

$$\langle S(w, \bar{w}) \rangle = \prod_{\alpha}^{N} Z_\alpha(v_\alpha) - \frac{1}{K}(X(w) - \hat{X}(w)) \tag{15}$$

where K is the cardinality i.e. $K = |\bar{W}|$. Then, the solution of the M step after plugging (15) in $\frac{\partial \mathcal{L}_{EU}}{\partial Z_\alpha(v_\alpha)}$ and by setting it to zero

$$\begin{aligned} \frac{\partial \mathcal{L}_{EU}}{\partial Z_\alpha(v_\alpha)} &= \sum_{v \notin V_\alpha} M(w) \left(\left(X(w) - \hat{X}(w) \right) \prod_{\alpha' \neq \alpha} Z_{\alpha'}(v_{\alpha'}) \right) \\ &= \Delta_\alpha(M \circ X) - \Delta_\alpha(M \circ \hat{X}) = 0 \end{aligned} \tag{16}$$

The solution of this fixed point equation leads to two related but different iterative schemata: multiplicative updates (MUR) and alternating least squares (ALS).

PLTF$_{EU}$ Multiplicative Update Rules (MUR). This method is indeed gradient ascent similar to [10] by setting $\eta(v_\alpha) = Z_\alpha(v_\alpha)/\Delta_\alpha(M \circ \hat{X})(v_\alpha)$ as

$$Z_\alpha(v_\alpha) \leftarrow Z_\alpha(v_\alpha) + \eta(v_\alpha) \frac{\partial \mathcal{L}_{EU}}{\partial Z_\alpha(v_\alpha)} \tag{17}$$

Then the update rule becomes simply

$$Z_\alpha \leftarrow Z_\alpha \circ \Delta_\alpha(M \circ X)/\Delta_\alpha(M \circ \hat{X}) \tag{18}$$

PLTF$_{EU}$ Alternating Least Squares (ALS). The idea behind ALS is to obtain a closed form solution for Z_α directly from (16)

$$\Delta_\alpha(M \circ X) = \Delta_\alpha(M \circ \hat{X}) \tag{19}$$

Note that $\hat{X}$ depends on Z_α, see (1). This equation can be solved for Z_α by least squares, as it is linear in Z_α. If there is no missing data ($M(w) = 1$ for all w), the result is available in closed form. To see this, we write all the tensors in matrix form and write the solution explicitly using standard matrix algebra.

3 Matricization

Matricization as defined in [8,9] is the operation of converting a multiway array into a matrix by reordering the column fibers. In this paper we refer to this definition as 'unfolding' and refer to *matricization* as the procedure to convert an element-wise equation (such as (19)) into a corresponding matrix form. We use Einstein's summation convention where repeated indices are added over.

The conversion rules are given in Table 1. Our notation is best illustrated with an example: consider a matrix $X^{i,j}$ with row index i and column index j. If we assume a column by column memory layout, we refer to the vectorisation of $\mathbf{vec}\, X$ (vertical concatenation of columns) as X_{ji}; adopting a 'faster index last' convention and we drop the commas. Here i is the faster index since when traversing the elements of the matrix X in sequence i changes more rapidly. With this, we arrive at the following definition:

Definition 2. *Consider a multiway array $X \in \mathbb{R}^{I_1 \times \ldots \times I_L}$ with a generic element denoted by $X^{i_1,i_2,\ldots,i_L}$. The mode-n unfolding of X is the matrix $X_{(n)} \in \mathbb{R}^{I_n \times \prod_{k \neq n} I_k}$ with row index i_n where*

$$X_{(n)} \equiv X_{i_n}^{i_L \ldots i_{n-1} i_{n+1} \ldots i_2 i_1} \tag{20}$$

Table 1. Index notation used to unfold a multiway array into the matrix form. Following Einstein convention, duplicate indices are summed over. Khatri-Rao product and mode-n unfolding are implemented in N-way Toolbox [2] as *krb*() and *nshape*().

Equivalance	Matlab	Remark
$X_i^j \equiv X$	X	Matrix notation
$X_i^{kj} \equiv X_{(1)}$	$nshape(X, 1)$	Array (mode-1 unfolding)
$X_i^j \equiv (X^T)_j^i$	X'	Transpose
$\mathbf{vec}\, X_i^j \equiv (X)_{ji}$	$X(:)$	Vectorize
$X_i^j Y_j^p \equiv (XY)_i^p$	$X * Y$	Matrix product
$X_i^p Y_j^p \equiv (X \odot Y)_{ij}^p$	$krb(X, Y)$	Khatri-Rao product
$X_i^p Y_j^q \equiv (X \otimes Y)_{ij}^{pq}$	$kron(X, Y)$	Kronecker product

In the following, we illustrate the derivation of the well known TUCKER3 factorization. Alternative models can be derived and implemented similarly. To save the space we only give derivations for factor A and core tensor G and omit the others. Further details, model examples and reference implementations in Matlab can be found from `http://www.sibnet.com.tr/pltf`.

Example 2 (Derivation of matrix form update rules for the TUCKER3 decomposition). We compute first the prediction in matrix form

$$\hat{X}^{i,j,k} = \sum_{pqr} G^{p,q,r} A^{i,p} B^{j,q} C^{k,r} \tag{21}$$

$$(\hat{X}_{(1)})_i^{kj} = (G_{(1)})_p^{rq} A_i^p B_j^q C_k^r = \left((AG_{(1)})((C \otimes B)^T)\right)_i^{kj} \tag{22}$$

$$\hat{X}_{(1)} = AG_{(1)}(C \otimes B)^T \tag{23}$$

Algorithm 1 *Probabilistic Latent Tensor Factorisation.*

for epoch $= 1 \dots$ MAXITER **do**
  for $\alpha = 1 \dots N$ **do**
    $\hat{X} \leftarrow \sum_{\bar{w} \in \bar{W}} \prod_{\alpha} Z_{\alpha}(V_{\alpha})$
    if KL: $Z_{\alpha} \leftarrow Z_{\alpha} \circ \Delta_{\alpha}(M \circ X / \hat{X}) / \Delta_{\alpha}(M)$
    if EUC-MUR: $Z_{\alpha} \leftarrow Z_{\alpha} \circ \Delta_{\alpha}(M \circ X) / \Delta_{\alpha}(M \circ \hat{X})$
    if EUC-ALS: Solve $\Delta_{\alpha}(M \circ X) = \Delta_{\alpha}(M \circ \sum_{\bar{w} \in \bar{W}} \prod_{\alpha} Z_{\alpha}(V_{\alpha}))$ for Z_{α}
  end for
end for

Now, Δ_{Z_α} for all α can also be represented in matrix form. The functions Δ_A and Δ_G are

$$\Delta_A(X) \equiv (X_{(1)})_i^{kj} B_j^q C_k^r G_p^{rq} \equiv X_{(1)}(C \otimes B) G_{(1)}^T \tag{24}$$

$$\Delta_G(X) \equiv (X_{(1)})_i^{kj} A_i^p B_j^q C_k^r \equiv A^T X_{(1)}(C \otimes B) \tag{25}$$

– if KL ((13)) we evaluate $\Delta_\alpha(Q)$ and $\Delta_\alpha(M)$ where $Q = M \circ (X/\hat{X})$

$$A \leftarrow A \circ \frac{Q_{(1)}(C \otimes B) G_{(1)}^T}{M_{(1)}(C \otimes B) G_{(1)}^T} \qquad G_{(1)} \leftarrow G_{(1)} \circ \frac{(A^T Q_{(1)})(C \otimes B)}{(A^T M_{(1)})(C \otimes B)} \tag{26}$$

– if EUC-ALS. We solve $\Delta_\alpha(X) = \Delta_\alpha(\hat{X})$ (19) when there are no missing observations, i.e., $M(w) = 1$ for all w. We show only the updates for the core tensor G. The pseudo-inverse of A is denoted by $A^\dagger$. From (25) we have

$$A^T X_{(1)}(C \otimes B) = A^T \left(A G_{(1)} (C \otimes B)^T \right) (C \otimes B) \tag{27}$$

$$G_{(1)} \leftarrow A^\dagger X_{(1)} \left((C \otimes B)^T \right)^\dagger \tag{28}$$

4 Discussion

The main saving in our framework appears in the computation of Δ_α, that is computationally equivalent to computing expectations under probability distributions that factorise according to a given graph structure. As is the case with graphical models, this quantity can be computed a-la belief propagation: algebraically we distribute the summation over all $v \notin V_\alpha$ and compute the sum in stages. For MUR, the intermediate computations carried out when computing the denominator and numerator can be reused, which leads to further savings (e.g., see (26)). Perhaps more importantly, PLTF encourages the researchers to 'invent' new factorization models appropriate to their applications. Pedagogically, the framework guides building new models as well as deriving update equations for KL and Euclidean cost functions. Indeed, results scattered in the literature can be derived in a straightforward manner.

Due to space constraints, in this paper we could not detail on model selection issues, i.e., questions regarding to the dimensions of latent indices and the

selection of an optimal factorisation structure, guided by data. It turns out, exploiting the probabilistic interpretation and choosing an appropriate prior, it is indeed possible to approximate the marginal likelihood $p(X) = \int dZ p(X, Z)$ for doing Bayesian model selection, using techniques described in [4].

References

1. E. Acar and B. Yener. Unsupervised multiway data analysis: A literature survey. *IEEE Transactions on Knowledge and Data Engineering*, 21:6–20, 2009.
2. C. A. Andersson and R. Bro. The N-way toolbox for MATLAB. *Chemometrics and Intelligent Laboratory Systems*, 52:1–4, 2000.
3. A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh. Clustering with Bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, 2005.
4. A. T. Cemgil. Bayesian inference for nonnegative matrix factorisation models. *Computational Intelligence and Neuroscience*, 2009:1–17, 2009.
5. A. Cichocki, R. Zdunek, A. H. Phan, and S. Amari. *Nonnegative Matrix and Tensor Factorization*. Wiley, 2009.
6. C. Fevotte and A. T. Cemgil. Nonnegative matrix factorisations as probabilistic inference in composite models. In *Proc. 17th EUSIPCO*, 2009.
7. M. I. Jordan, editor. *Learning in Graphical Models*. MIT Press, 1999.
8. H. A. L. Kiers. Towards a standardized notation and terminology in multiway analysis. *Journal of Chemometrics*, 14:105–122, 2000.
9. T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.
10. D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *NIPS*, volume 13, pages 556–562, 2001.
11. P. L. Meyer. *Introductory Probability and Statistical Applications*. Reading MA, Addison-Wesley, 2nd edition, 1970.
12. M. Mørup, L. K. Hansen, and S. M. Arnfred. Algorithms for sparse non-negative TUCKER. *Neural Computation*, 20(8):2112–2131, 2008.
13. R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, volume 20, 2008.
14. M. Schmidt and S. Mohamed. Probabilistic non-negative tensor factorisation using Markov Chain Monte Carlo. In *17th European Signal Processing Conference*. 2009.
15. A. P. Singh and G. J. Gordon. A unified view of matrix factorization models. In *ECML PKDD'08, Part II*, number 5212, pages 358–373. Springer, 2008.