Point-by-Point Response Letter

Dear Professor Anandkumar,

We greatly appreciate all the constructive comments from you and three anonymous referees. Following your suggestions, we have thoroughly revised the manuscript, and carefully addressed all the questions. We believe this has led to a much improved paper, and hope you find this revision satisfactory. In the revised manuscript, we have marked the changes in red color for easy reference. In the response letter, we first summarize the major changes, then provide our point-by-point responses to all comments.

1. **Novelty and major contributions**: We have added a new **Section 1.3** to clarify the main contributions that set our work apart from the existing literature. First, we systematically quantify the hardness of the binary tensor decomposition problem, by characterizing the impact of latent signal-to-noise ratio (SNR) on the tensor recovery accuracy, and identifying three different phases for tensor recovery according to SNR. Second, we propose a new method for binary tensor decomposition and establish the statistical properties of the global optimizer, including the upper bound and the minimax lower bound on the tensor recovery accuracy. Third, we have added the algorithmic convergence properties for the actual optimizer from our block relaxation estimation algorithm. Please see our responses to Referee 1, Point 1, and Referee 3, Point 1, for more details.

2. **Literature review**: We have added more references in **Section 1.2**. In addition to the related work reviewed in the first version of the paper, we have now added more discussion on higher-order binary tensor decomposition and Boolean tensor decomposition, and clarify the difference between our method and those solutions. Please see our responses to Referee 1, Point 1, and Referee 3, Point 2, for more details.

3. **Algorithmic properties**: In the first version of this manuscript, we primarily focused on the statistical properties of the global constrained maximum likelihood estimator, which characterize the intrinsic population optimality of the estimator and are independent of any specific algorithm. In this revision, we have added a new **Section 4.2** to study the algorithmic convergence properties. We establish an algorithm-dependent error bound for the actual optimizer, which reveals an interesting interplay between the computational efficiency and the statistical convergence. Please see our response to Referee 2, Point 1, for more details.

4. **Numerical simulations and comparison**: Following the reviewers' suggestions, we have expanded our numerical experiments considerably. Specifically, we have added new simulations in **Section 5.1**, by considering the tensor dimension $d \in \{20, 40, 60\}$ and rank $R \in \{5, 10, 20, 40\}$. This way, in some of the combinations, the rank equals or exceeds the tensor dimension. We have also added a new **Section 5.3** to numerically compare our method with three existing solutions, Boolean tensor factorization, Bayesian tensor factorization, and Bernoulli tensor factorization with gradient descent. Please see our response to Referee 1, Points 2 and 3, Referee 3, Point 3, for more details.

Point-by-point Responses to Referee 1

We greatly appreciate your valuable comments and suggestions. We have carefully addressed all the questions. In the revised manuscript, we have marked the major changes in red color for easy reference. In this response letter, your comments are shown in *italics*, followed by our point-by-point responses.

This paper presents a modification of the continuous data, Frobenius norm, CP tensor decomposition, which is appropriate for binary data by adopting Bernoulli modeling of the observed tensor entries. Subsequently, the paper presents performance bounds for the proposed model.

The paper is well-written, and the theoretical analysis appears to be sound. However, the novelty of the present paper lies perhaps solely on said theoretical analysis and bound derivations, in light of pertinent related work that is currently not acknowledged in the manuscript.

Response: Thank you for your encouraging feedback. We address your question about the novelty in detail below.

1. *As mentioned above, "special" treatment of binary values in a tensor is by no means a new problem, and there exists related work that addresses exactly that problem.*

   The earliest work, to the best of the reviewer's knowledge, is [a], where the author proposes a boolean tensor factorization, where additions and multiplications are replaced by logical operations.

   Most recently, [b] have introduced, as an upcoming part of the tensor toolbox for Matlab, a generalized CP decomposition, where different loss functions are used, as a result of modeling the data using different distributions; amongst them, Bernoulli is one such distribution, and Laplace, which accounts for count data, a topic of future work for this paper under-review (which has been extensively studied in [c]).

   In light of all the existing work, i) there should be a thorough description of them in the related work section, and a clear discussion on what and why is new here, besides the derived bounds, and ii) inclusion of those approaches in the experimental evaluation/comparisons.

   Response: Thank you very much for pointing out those relevant papers, and for the suggestions on how to improve the paper. The reference [b] was already included in the first version, but we have added a more detailed discussion in the revision. We have added more relevant papers suggested by you and another referee. Moreover, we have further clarified our contributions and have added new simulations to compare our method with the existing solutions.

   Literature review: We agree with you that the same problem of binary tensor decomposition has been studied by earlier papers. We have added a more detailed discussion in **Section 1.2, Related work.**

   In particular, one closely related line of work is higher-order binary tensor decomposition recently studied by [13, 16, 10] (among which [10] is [b] you mentioned). We target the same problem. However, our study differs in terms of the scope of the results. In

general, there are two types of properties that an estimator possesses. The first type is the algorithm-dependent property that quantifies the impact of a specific algorithm, such as the choice of loss function, initialization, and iterations, on the final estimator. The second type is the statistical property that characterizes the population behavior and is independent of any specific algorithm. Earlier solutions of [13, 16, 10] focused only on the algorithm effectiveness, but did not address the population optimality. By contrast, we address both algorithmic and statistical aspects of the problem. We study the statistical properties of the global optimizer in Section 3, and add a new Section 4.2 to study the algorithmic properties of the actual optimizer of our estimation algorithm. Studying both types of properties allows us to better understand the gap between a specific algorithm and the population optimality, which may in turn offer a useful guide to the algorithm design. This is one of the key differences between our paper and the earlier solutions.

Another relevant line of work is Boolean tensor decomposition developed by [14, 6, 18, 11] (among which [14] is [a] you mentioned). This is a family of data-driven algorithms that use logic operations to decompose a binary tensor into binary factors. These methods also dealt with binary tensor, same as we do, but they took a model-free approach to approximate the data instance. One important difference is that we focus on parameter estimation in a population model. The population interpretation offers useful insight on the effectiveness of dimension reduction. In addition, having a population model allows us to tease apart the algorithmic error versus statistical error. In this respect, our proposal is very different from boolean tensor decomposition. We also numerically compare the two approaches in our newly added simulations. Please see our response below.

Contributions: We have added a new **Section 1.3** to further clarify the main contributions that set our work apart from the existing literature.

First, we systematically quantify the hardness of the binary tensor decomposition problem. We show that the Bernoulli tensor model (1) is equivalent to entrywise quantization of a latent noisy continuous-valued tensor. We then characterize the impact of latent signal-to-noise ratio (SNR) on the tensor recovery accuracy, and identify three different phases for tensor recovery according to SNR; see Table 1 in Section 3.3. When SNR is bounded by a constant, the loss in binary tensor decomposition is comparable to the case of continuous-valued tensor, suggesting very little information has been lost by quantization. On the other hand, when SNR is sufficiently large, stochastic noise turns out to be helpful, and is in fact essential, for estimating the signal tensor. The later effect is related to "dithering" [5] and "perfect separation" [2] phenomenon, and is clearly contrary to the behavior of continuous-valued tensor decomposition.

Second, we propose a new method for binary tensor decomposition and establish its statistical properties, including the upper bound and the minimax lower bound on the tensor recovery accuracy. These properties characterize the intrinsic population optimality of the estimator and are independent of any specific algorithm. Note that, in our problem, the tensor dimensions $(d_1, \ldots, d_K)$ diverge along with the sample size, and so does the number of unknown parameters. As such, the classical maximum likelihood estimation (MLE) theory does not directly apply. We leverage the recent development in random tensor theory and high-dimensional statistics, and establish

the error bounds of the tensor estimation. The matching information-theoretical lower bounds are correspondingly provided. In particular, when the tensor dimensions are the same in all modes, $d_1 = \ldots = d_K = d$, we obtain a convergence rate $\asymp d^{-K+1}$ for estimating Θ. This rate outperforms the rate of "best matrixization", which is $\asymp d^{-\lfloor K/2 \rfloor}$, and $\lfloor K/2 \rfloor$ is the integer part of $K/2$, as well as the rate of 1-bit tensor completion, which is $\asymp d^{-(K-1)/4}$ [8]. To our knowledge, these statistical guarantees are among the first for binary tensor decomposition.

Lastly, we supplement the general statistical properties by proposing a block relaxation algorithm, and establish the corresponding algorithmic convergence properties. Our algorithm-dependent error bound reveals an interesting interplay between the computational efficiency and the statistical convergence. We also illustrate the efficacy of our algorithm through both simulations and real data applications.

Numerical comparison: Following your suggestion, we have now added a new **Section 5.3** to numerically compare our method with the existing solutions.

Specifically, we compare with the following alternative solutions for binary tensor factorization (BTF).

- Boolean tensor factorization (BooleanTF) [14, 7, 18]. This method decomposes a binary tensor into binary factors, then cover the binary entries based on a set of logic rules among the factors. We use the implementation of [18].
- Bayesian tensor factorization (BTF_Bayeisan) [17]. This method uses the expectation-maximization approach to decompose a binary tensor into a number of continuous-valued factors. It imposes a Gaussian prior on the factor entries, and a multiplicative gamma process prior on the factor weights $\{\lambda_r\}$.
- Bernoulli tensor factorization with gradient descent (BTF_Gradient) [10]. This method uses a gradient descent algorithm to decompose a binary tensor into continuous-valued factors. We use the implementation in the toolbox of Matlab.

For easy reference, we denote our method by BTF_Alternating, and our implementation can be found at https://github.com/Miaoyanwang/Binary-Tensor. These four methods differ in several ways. BooleanTF is different from the other three in both the cost function and the output format. The rest are all based on the Bernoulli model (2), but with different implementations. BTF_Bayesian employs a Bayesian approach, whereas the other two are frequentist solutions. BTF_Gradient and our method, BTF_Alternating, share the same model, but utilize different optimization algorithms. So the two methods complement each other. On the other hand, in this article, we provide both the algorithm-specific convergence properties, and the algorithm-independent statistical properties including the statistical convergence rate, SNR phase diagram, and mini-max rate. These results are not available in [10] who proposed BTF_Gradient, but these properties hold for BTF_Gradient as well.

We apply the four methods with the default parameters, while selecting the rank R using the recommended approach of each. For our method BTF_Alternating, we use the proposed BIC to select the rank. Since BTF_Gradient does not provide any rank selection criterion, we apply the same R selected by our BIC. For BTF_Alternating, we also set the hyper-parameter α to infinity, which essentially poses no prior on the

tensor magnitude. Besides, because BTF_Bayesian only supports the logistic link, we use the logistic link in all three BTF methods.

We evaluate each method by two metrics. The first metric is the root mean square error, RMSE $= \left(\sqrt{\prod_k d_k}\right)^{-1} \|\widehat{\mathbb{E}(\mathcal{Y})} - \mathbb{E}(\mathcal{Y})\|_F$, where $\widehat{\mathbb{E}(\mathcal{Y})}$ denotes the estimated success probability tensor. For BooleanTF, this quantity is represented as the posterior mean of $\mathcal{Y}$ [14], and for the other three methods, $\widehat{\mathbb{E}(\mathcal{Y})} = \text{logit}(\hat{\Theta})$. The second metic is the misclassification error rate, MER $= \left(\prod_k d_k\right)^{-1} \|\mathbb{1}_{\widehat{\mathbb{E}(\mathcal{Y})} \geq 0.5} - \mathbb{1}_{\mathbb{E}(\mathcal{Y}) \geq 0.5}\|_0$. Here the indicator function is applied to tensors in an element-wise manner, and $\|\cdot\|_0$ counts the number of non-zero entries in the tensor. These metrics reflect two aspects of the construction error. RMSE summarizes the total estimation error in the success probabilities, whereas MER summarizes the classification errors among 0's and 1's.

We simulate data from two different models, and in both cases, the signal tensors do not necessarily follow an exact low-rank CP structure. Therefore, in addition to method comparison, it also allows us to evaluate the robustness of our method under potential model misspecification. The results are reported in Figure 4 of the paper. For easy reference, we reproduce the plot here in Figure A in the response letter.

The first model is a logic boolean tensor model following the setup in [18]. We first simulate noiseless tensors $\mathcal{Y} = [\![y_{ijk}]\!]$ from the following model,

$$y_{ijk} = \bigvee_{r=1}^{R} \bigwedge_{ijk} a_{ir} b_{jr} c_{kr},\ a_{ir} \sim \text{Ber}(p^a_{ir}),\ b_{jr} \sim \text{Bernoulli}(p^b_{jr}), c_{kr} \sim \text{Bernoulli}(p^c_{kr}),$$

where the binary factor entries $\{a_{ir}\}$, $\{b_{jr}\}$, $\{c_{kr}\}$ are mutually independent with each other, the factor probabilities $\{p^a_{ir}\}$, $\{p^b_{jr}\}$, $\{p^c_{kr}\}$ are generated i.i.d. from Beta(2,4), and $\vee$ and $\wedge$ denote the logical OR and AND operations, respectively. Equivalently, the tensor entry is 1 if and only if there exists one or more components in which all corresponding factor entries are 1. It is easy to verify that

$$\mathbb{E}(y_{ijk} | \{p^a_{ir}, p^b_{jr}, p^c_{kr}\}) = 1 - \prod_{r=1}^{R} \left(1 - p^a_{ir} p^b_{jr} p^c_{kr}\right).$$

We then add contamination bias to $\mathcal{Y}$ by flipping the tensor entries $0 \leftrightarrow 1$ i.i.d. with probability 0.1. We consider the tensor dimension $d_1 = d_2 = d_3 = 50$, and the boolean rank $R \in \{10, 15, 20, 25, 30\}$.

Figure A(a)-(b) shows the performance comparison based on $n_{\text{sim}} = 30$ replications. It is seen that the three BTF methods outperform BooleanTF in RMSE. This shows the advantage of a probabilistic model, upon which all three BTF methods are built. In contrast, BooleanTF seeks patterns in a specific data, but does not address population estimation. For classification, BooleanTF performs reasonably well in distinguishing 0's versus 1's, which agrees with the data mining nature of BoolenTF. It is also interesting to see that MER peaks at $R = 20$. Further investigation reveals that this setting corresponds to the case when the Bernoulli probabilities $\mathbb{E}(\mathcal{Y})$ concentrate around 0.5, which becomes particularly challenging for classification. Actually, the average Bernoulli probability for R =10, 15, 20, 25, 30 is 0.31, 0.44, 0.53, 0.61, 0.68, respectively. Figure A(b) also shows that BTF_Alternating and BTF_Gradient achieve

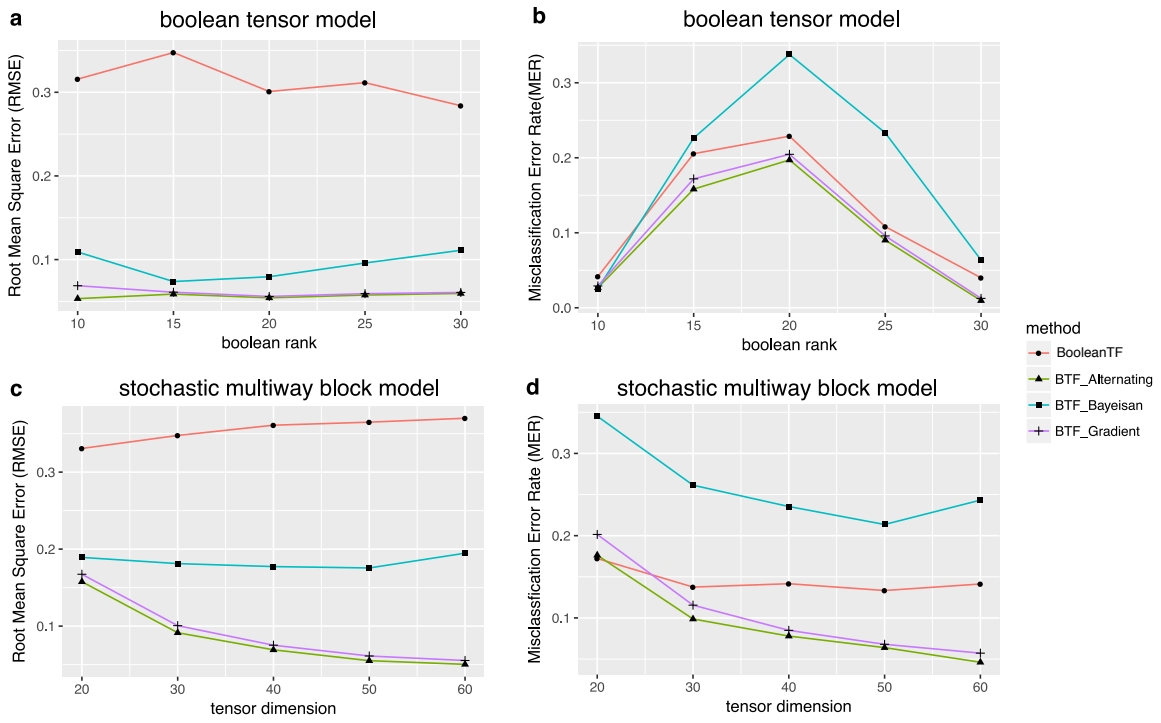

Figure A: Performance comparison in terms of root mean squared error and misclassification error rate. (a)-(b) Estimation errors for the boolean tensor model. (c)-(d) Estimation errors for the stochastic multiway block model.

a smaller classification error than BTF_Bayesian. One possible explanation is that the normal prior in BTF_Bayesian has a poor distinguishing power around $\theta \approx 0$, which corresponds to the hardest case when Bernoulli probability ≈ 0.5.

The second model is the stochastic multi-way block model considered in Section 5.2, with the block means $\{c_{m_1 m_2 m_3}\}$ generated from the combinatorial-mean sub-model. Figure A(c)-(d) shows the performance comparison, and a similar pattern is observed. The two frequentist-type BTF methods, BTF_Gradient and BTF_Alternating, behave numerically similarly, and they outperform the other alternatives. In particular, the BTF methods exhibit decaying estimation errors, whereas BooleanTF appears to flatten out as dimension grows. This observation suggests that statistical error is likely more dominating in this setting.

2. *For rank selection, the current paper proposes using the BIC criterion. How does that compare against well-known methods for selecting the rank of a tensor, such as [d,e,f]? If this is a contribution of the paper, then there should be a clear justification on why this is the best approach and why this is novel, and there should be thorough comparisons against standard methods in the experimental section. If, on the other hand, rank selection is not a contribution of the paper, then perhaps the experiments on it are redundant and definitely incomplete.*

   It is also unclear how the rank estimation is actually conducted. Is the BIC minimized over grid search? If so, how many ranks, higher than the true one, are tested? This could reveal how prone is the criterion to overfactoring.

	$\sigma = 0.1$			$\sigma = 0.01$		
True rank	$d = 20$	$d = 40$	$d = 60$	$d = 20$	$d = 40$	$d = 60$
$R = 5$	4.9 (0.2)	5 (0)	5 (0)	4.8 (1.0)	5 (0)	5 (0)
$R = 10$	8.7 (0.9)	10 (0)	10 (0)	8.8 (0.4)	10 (0)	10 (0)
$R = 20$	17.7(1.7)	20.4(0.5)	20.2(0.5)	16.4(0.5)	20.4(0.5)	20.6(0.5)
$R = 40$	36.8(1.1)	39.6(1.7)	40.2(0.4)	36.0(1.2)	38.8(1.6)	40.3(1.1)

Table A: Rank selection in binary tensor decomposition via BIC. The selected rank is averaged across 30 simulations, with the standard error shown in the parenthesis.

Furthermore, in the experimental evaluation of rank selection, it would be very interesting to test the case where the rank is higher than the dimension of the smallest mode. This is the most challenging case in selecting the rank and could demonstrate advantages of the proposed approach.

Finally, on page 18, in the description of the results for the real data, there is a major inconsistency which is not explained. BIC gives 9 components and elbow method gives 5. Which one is the right answer? Why opt in for the highest number? How likely is it that BIC gives an overestimation of the real rank? This point here should be thoroughly discussed or removed because right now it is basically stating that two heuristics were tried and the proposed one was chosen, but without any justification about its plausible correctness.

Response: Thank you for these questions about the rank selection. We have further clarified about our BIC criterion in the revision.

Relation to other selection criterion: Our BIC criterion is a statistical approach for selecting the rank in a population model. It aims to balance between the goodness-of-fit for the data and the degree of freedom in the population model. In contrast, the selection methods in [d,e,f] were designed for rank selection for a particular dataset. Moreover, there is another minor difference, in that we aim at CP tensor decomposition R, whereas the multilinear rank [d,e,f] targets Tucker tensor decomposition. We have added a comment in **Section 4.3, Page 16, Line -9 to -8**.

Implementation: We have clarified about our implementation on rank selection in **Section 5.1, Page 18, Lines 5 to 6**. Specifically, we minimize BIC using a grid search from $R - 5$ to $R + 5$.

More simulations: We have added new simulations in **Section 5.1**, by considering the tensor dimension $d \in \{20, 40, 60\}$ and rank $R \in \{5, 10, 20, 40\}$. This way, in some of the combinations, the rank equals or exceeds the tensor dimension. We report the average selection results over $n_{\text{sim}} = 30$ replications in Table 2 of the paper. For easy reference, we reproduce the results in Table A here. It is seen that, when $d = 20$, the selected rank is close to but slightly smaller than the true rank, for both small R and large R. As d increases, the rank selection becomes more accurate, since in tensor decomposition, the total number of entries corresponds to the sample size, so a larger d implies a larger sample size.

Data analysis: Sorry for the confusion. Regarding *"BIC gives 9 components and elbow method gives 5"*, this is not what we intended to convey.

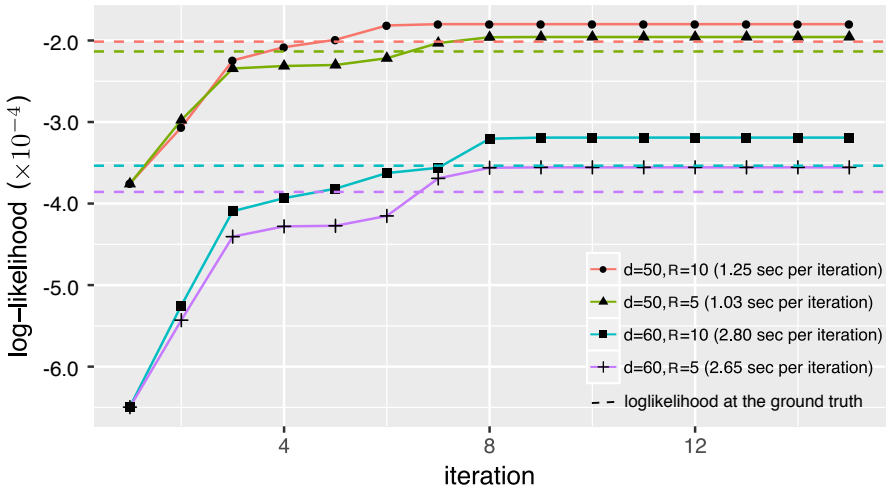

Figure B: Trajectory of the objective function over iterations with varying d and R.

In the real data analysis, we have utilized a two-step procedure, by first applying the proposed binary tensor decomposition method with the logistic link, then applying the K-means clustering along the country mode from the decomposition. In the first step, the BIC criterion suggests $R = 9$ factors, and in the second step, the classical elbow method selects 5 clusters. So there is no inconsistency in selection, since they were for two different selection problems, one for the tensor rank selection, and the other for the number of clusters selection.

We realize that the original description may cause confusion, so we have added the above clarification in **Section 6, Page 24, Lines 5 to 10.**

3. *In the simulations, it is mentioned that "we often find that the converged points have nearly optimal objective values". How is this claim possible? Is the optimal objective value known? If so, what is it and how was it computed? If not, then this is an empirical claim and should be clearly stated as such. There should be a clarification on how this conclusion is reached (e.g., running 100 different iterations, 95% of them converged to the same point, therefore we believe that this must be the optimal, or something along these lines).*

Response: Thank you for pointing this out! You are right that the optimal objective value is unknown. On the other hand, we can obtain the objective value at the true parameter, i.e., $\mathcal{L}_{\mathcal{Y}}(\Theta_{\text{true}})$, in our simulations. We have now rephrased our statement in **Section 5.1, the second paragraph on Page 18**, by saying that, "Although Algorithm 1 has no theoretical guarantee to land to the global optimum, in practice, we often find that the converged points are satisfactory, in that the corresponding objective values are close to the objective function evaluated at the true parameter, $\mathcal{L}_{\mathcal{Y}}(\Theta_{\text{true}})$. As an illustration, Figure 3 shows the typical trajectories of the objective function under different tensor dimensions and ranks. The dashed line is the objective value at the true parameter, $\mathcal{L}_{\mathcal{Y}}(\Theta_{\text{true}})$. It is seen that the algorithm generally converges quickly upon random initializations, usually taking fewer than 8 iterations for the relative change in the objective to be below 3%, even for a large d and R." We have also reproduced the plot here in Figure B for easy reference.

Point-by-point Responses to Referee 2

We greatly appreciate your valuable comments and suggestions. We have carefully addressed all the questions. In the revised manuscript, we have marked the major changes in red color for easy reference. In this response letter, your comments are shown in *italics*, followed by our point-by-point responses.

1. *This paper combine the ideas of general linear model (GLM) and tensor decomposition (TD) to solve the problem of unsupervised learning in binary tensors. (1) From the algorithmic perspective, the paper provides an elegant ALS-like alternative maximization (AM) algorithm to recover the latent factors given a binary tensor; and (2) from the statistical perspective, the paper proves the rate-optimality of maximum likelihood estimator (MLE) for the proposed model.*

   There is a huge gap between these two perspectives: (1) Under what condition the suggested algorithm converges to the global minimum, i.e. the MLE? (2a) If the global convergence is not guaranteed, whether the algorithm converges to a local minimum (or converges to the border of the feasible set D)? (2b) If the algorithm converges to local minimum, is there any statistical guarantee for the sub-optimal estimator? (3) What is the convergence rate (either global or local) for the suggested algorithm?

   Since (1) AM algorithm for non-convex optimization and (2) rate-optimality of MLE in Statistics are both well-known (although they may not have been developed for the specific setting of probabilistic tensor decomposition), I think the most important but unresolved problem is how the algorithm achieves statistical optimality (which is the key difference between machine learning and theoretical statistics).

   Response: We greatly appreciate this question. In the revision, we have first clarified that there are two types of properties that an estimator possesses. The first type is the algorithm-dependent property that quantifies the impact of a specific algorithm, such as the choice of loss function, initialization, and iterations, on the final estimator. The second type is the statistical property that characterizes the population behavior and is independent of any specific algorithm. We now study both types of properties, which sets our work apart from the existing literature on binary tensor decomposition that mostly focused on the first type of properties.

   In the first version of this paper, we primarily focused on the statistical properties of the global constrained maximum likelihood estimator, including the upper bound and the minimax lower bound on the tensor recovery accuracy. These properties characterize the intrinsic population optimality of the estimator and are independent of any specific algorithm. We also systematically quantify the hardness of the binary tensor decomposition problem. We first show that the Bernoulli tensor model is equivalent to entrywise quantization of a latent noisy continuous-valued tensor. We then characterize the impact of latent signal-to-noise ratio (SNR) on the tensor recovery accuracy, and identify three different phases for tensor recovery according to SNR. We have added the above clarification in the new **Section 1.3**.

   In this revision, following your suggestion, we have now added a new **Section 4.2** to study the algorithmic convergence properties. We establish an algorithm-dependent

error bound for the actual optimizer, which reveals an interesting interplay between the computational efficiency and the statistical convergence.

More specifically, because the block relaxation algorithm monotonically increases the objective function, the convergence of the objective function is guaranteed whenever the $\mathcal{L}$ is bounded from above. We study the convergence of the actual iterates $\boldsymbol{A}^{(t)}$ and $\Theta^{(t)} = \Theta\{\boldsymbol{A}^{(t)}\}$ resulting from Algorithm 1. To simplify the analysis, we set the hyper-parameter α to infinity, which essentially poses no prior on the tensor magnitude. We need the following assumptions.

(A1) (Regularity condition) The log-likelihood $\mathcal{L}(\boldsymbol{A})$ is continuous and the set $\{\boldsymbol{A}\colon \mathcal{L}(\boldsymbol{A}) \geq \mathcal{L}(\boldsymbol{A}^{(0)})\}$ is compact.

(A2) (Strictly local maximum condition) Each block update in Algorithm 1 is well-defined; i.e., the GLM solution exists and is unique, and the corresponding sub-block in the Hession is non-singular at the solution.

(A3) (Local uniqueness condition) The set of stationary points of $\mathcal{L}(\boldsymbol{A})$ are isolated and module scaling.

(A4) (Local Lipschitz condition) The tensor rank-R representation $\Theta = \Theta(\boldsymbol{A})$ is called locally Lipschitz at $\boldsymbol{A}^*$, if there exits two constants $c_1, c_2 > 0$ such that

$$c_1\|\boldsymbol{A}' - \boldsymbol{A}''\|_F \leq \|\Theta(\boldsymbol{A}') - \Theta(\boldsymbol{A}'')\|_F \leq c_2\|\boldsymbol{A}' - \boldsymbol{A}''\|_F,$$

for $\boldsymbol{A}', \boldsymbol{A}''$ sufficiently close to $\boldsymbol{A}^*$. Here $\boldsymbol{A}', \boldsymbol{A}''$ represent the block variables subject to convention (10).

All these are fairly mild conditions and are often imposed in the literature. Specifically, Assumption (A1) ensures that the maximum exists and the log-likelihood is bounded above. Therefore, the stopping rule of Algorithm 1 is well defined. Assumption (A2) asserts the negative-definiteness of the Hessian in the block coordinate $\boldsymbol{A}_k$. Note that the full Hession needs not to be negative-definite in all variables simultaneously. We consider this as a reasonably mild assumption, and similar conditions have often been imposed in numerous non-convex problems [20, 21, 19]. Assumptions (A2)–(A4) guarantee the local uniqueness of the CP decomposition $\Theta = \Theta(\boldsymbol{A})$. The conditions exclude the case of rank-degeneracy; e.g., if the tensor Θ can be written in fewer than R factors, or if the columns of $\boldsymbol{A}_{-k}^{(t)}$ are linearly dependent in the GLM update. They also exclude the case of non-unique decompositions; e.g., if the decomposition Θ can be smoothly changed beyond scaling of the factors. These conditions are fairly mild for tensors of order 3 or higher. For more discussion on decomposition uniqueness and its implication in the optimization, we refer to [12, 20, 21].

Proposition 1 (Algorithmic convergence)**.** *Suppose Assumptions (A1)-(A3) hold.*

(i) (Global convergence) Any sequence $\boldsymbol{A}^{(t)} = \left\{\boldsymbol{A}_1^{(t)}, \ldots, \boldsymbol{A}_K^{(t)}\right\}$ *generated by Algorithm 1 converges to a stationary point of* $\mathcal{L}(\boldsymbol{A})$.

(ii) (Locally linear convergence) Let $\boldsymbol{A}^*$ *be a local maximizer of* $\mathcal{L}$. *There exists an* ε*-neighborhood of* $\boldsymbol{A}^*$, *such that, for any staring point* $\boldsymbol{A}^{(0)}$ *in this neighborhood, the iterates* $\boldsymbol{A}^{(t)}$ *of Algorithm 1 linearly converge to* $\boldsymbol{A}^*$,

$$\|\boldsymbol{A}^{(t)} - \boldsymbol{A}^*\|_F \leq \rho^t\|\boldsymbol{A}^{(0)} - \boldsymbol{A}^*\|_F,$$

where $\rho \in (0,1)$ is a contraction parameter. Furthermore, if Assumption (A4) holds at $\boldsymbol{A}^*$, then there exists a constant $C > 0$ such that

$$\|\Theta(\boldsymbol{A}^{(t)}) - \Theta(\boldsymbol{A}^*)\|_F \leq C\rho^t \|\Theta(\boldsymbol{A}^{(0)}) - \Theta(\boldsymbol{A}^*)\|_F.$$

Proposition 1(ii) shows that every local maximizer of $\mathcal{L}$ is an attractor of Algorithm 1. This is a nice property that ensures the exponential decay of the error near a local maximum. Moreover, it reflects some intrinsic difference between tensor decomposition and matrix decomposition, in that the same property often fails for the matrix case. Consider an example of a 2-by-2 matrix. Suppose that the local maximizer is $\Theta^* = \Theta^*(\boldsymbol{e}_1, \boldsymbol{e}_2) = \boldsymbol{e}_1^{\otimes 2} + \boldsymbol{e}_2^{\otimes 2}$, where $\boldsymbol{e}_1, \boldsymbol{e}_2$ are canonical vectors in $\mathbb{R}^2$. There is no region of attraction near the block variable $\boldsymbol{A}^* = (\boldsymbol{e}_1, \boldsymbol{e}_2)$. In fact, one can construct a point $\boldsymbol{A}^{(0)} = (\boldsymbol{a}_1, \boldsymbol{a}_2)$, with $\boldsymbol{a}_1 = (\sin\theta, \cos\theta)'$, and $\boldsymbol{a}_2 = (\cos\theta, -\sin\theta)'$. The point $\boldsymbol{A}^{(0)}$ can be made arbitrarily close to $\boldsymbol{A}^*$ by tuning θ, but the algorithm iteration initialized from $\boldsymbol{A}^{(0)}$ would never converge to $\boldsymbol{A}^*$. This behavior occurs when the matrix has degenerate singular values. In contract, a 2-by-2-by-2 tensor problem with the maximizer $\tilde{\Theta}^* = \tilde{\Theta}^*(\boldsymbol{e}_1, \boldsymbol{e}_2) = \boldsymbol{e}_1^{\otimes 3} + \boldsymbol{e}_2^{\otimes 3}$ possesses locally unique decomposition, and thus isolated stationary points. The block variable $\boldsymbol{A}^*$ exhibits a basin of attraction with respect to the tensor objective.

Combining Proposition 1 and Theorem 1, we have the following theorem.

Theorem 4 (Empirical performance)**.** *Suppose $\mathcal{Y} \in \{0,1\}^{d_1 \times \cdots \times d_K}$ is a binary tensor under the Bernoulli tensor model (2) with parameter $\Theta_{true} = \Theta(\boldsymbol{A}_{true})$. Let $\boldsymbol{A}^{(t)}$ denote a sequence of estimators generated from Algorithm 1, with the initial point $\boldsymbol{A}^{(0)}$ and the limiting point $\boldsymbol{A}^*$. Suppose that the initialization error $Loss(\Theta(\boldsymbol{A}^{(0)}), \Theta_{true})$ is bounded by some constant $\varepsilon > 0$, and that $\boldsymbol{A}^*$ satisfies $\mathcal{L}[\Theta(\boldsymbol{A}^*)] \geq \mathcal{L}(\Theta_{true})$. Suppose Assumptions (A1)-(A4) hold. Then we have, with probability at least $1 - exp(-C' \log K \sum_k d_k)$,*

$$Loss\left(\Theta(\boldsymbol{A}^{(t)}), \Theta_{true}\right) \leq \underbrace{C_1 \rho^t \varepsilon}_{\text{algorithmic error}} + \underbrace{\frac{C_2 L_\alpha}{\gamma_\alpha} \sqrt{\frac{R^{K-1} \sum_k d_k}{\prod_k d_k}}}_{\text{statistical error}}, \tag{A1}$$

where $\rho \in (0,1)$ is a contraction parameter, and $C_1, C_2 > 0$ are two constants.

Theorem 4 provides the estimation error of the actual estimator from our Algorithm 1 at each iteration. The bound (A1) consists of two terms: the first term is the computational error, and the second is the statistical error. The computational error decays exponentially with the number of iterations, whereas the statistical error remains the same as t grows. The statistical error is unavoidable, as it reflects the intrinsic error due to estimating from a noisy tensor; see also Theorem 2. The bound (A1) thus reveals the interplay between the computational and statistical errors. Moreover, for tensors with $d_1 = \ldots = d_K = d$, when the iteration number satisfies that,

$$t \geq T = \log_{1/\rho}\left(\frac{C_1 \varepsilon}{\frac{C_2 L_\alpha}{\gamma_\alpha}\sqrt{\frac{R^{K-1}\sum_k d_k}{\prod_k d_k}}}\right) \asymp \log_{1/\rho}\left\{d^{(k-1)/2}\right\},$$

the computational error is to be dominated by the statistical error.

For a more clear presentation, we have now re-organized Sections 3 and 4. That is, we present the statistical properties of the global optimizer, in terms of the upper and lower bounds and the phase-transition, in Section 3 of the paper. We then present the alternating updating algorithm and establish its algorithmic convergence properties in Section 4.

Point-by-point Responses to Referee 3

We greatly appreciate your valuable comments and suggestions. We have carefully addressed all the questions. In the revised manuscript, we have marked the major changes in red color for easy reference. In this response letter, your comments are shown in *italics*, followed by our point-by-point responses.

Major comments:

1. *The paper tackles the problem of binary tensor completion. Instead of reconstructing directly a binary tensor, the authors assume its entries are the realisation of independent Bernoulli variables. This continuous tensor is then decomposed using CP via Alternating optimization (one factor at a time) and line-search. Lower and upper bounds are provided. BIC is used to select rank.*

   The framework is overall interesting and the method also supports missing values. My main issue is that the novelty is low. Binary tensor decomposition was already proposed in previous work. Similarly, decomposition via block relaxation, and using BIC for rank selection are well known in the literature.

   Response: Thank you for a concise summary, and your question about the novelty. We have added a new **Section 1.3** to further clarify the main contributions that set our work apart from the existing literature.

   First, we systematically quantify the hardness of the binary tensor decomposition problem. We show that the Bernoulli tensor model (1) is equivalent to entrywise quantization of a latent noisy continuous-valued tensor. We then characterize the impact of latent signal-to-noise ratio (SNR) on the tensor recovery accuracy, and identify three different phases for tensor recovery according to SNR; see Table 1 in Section 3.3. When SNR is bounded by a constant, the loss in binary tensor decomposition is comparable to the case of continuous-valued tensor, suggesting very little information has been lost by quantization. On the other hand, when SNR is sufficiently large, stochastic noise turns out to be helpful, and is in fact essential, for estimating the signal tensor. The later effect is related to "dithering" [5] and "perfect separation" [2] phenomenon, and is clearly contrary to the behavior of continuous-valued tensor decomposition.

   Second, we propose a new method for binary tensor decomposition and establish its statistical properties, including the upper bound and the minimax lower bound on the tensor recovery accuracy. These properties characterize the intrinsic population optimality of the estimator and are independent of any specific algorithm. Note that, in our problem, the tensor dimensions $(d_1, \ldots, d_K)$ diverge along with the sample size, and so does the number of unknown parameters. As such, the classical maximum likelihood estimation (MLE) theory does not directly apply. We leverage the recent development in random tensor theory and high-dimensional statistics, and establish the error bounds of the tensor estimation. The matching information-theoretical lower bounds are correspondingly provided. In particular, when the tensor dimensions are the same in all modes, $d_1 = \ldots = d_K = d$, we obtain a convergence rate $\asymp d^{-K+1}$ for estimating Θ. This rate outperforms the rate of "best matrixization", which is $\asymp d^{-\lfloor K/2 \rfloor}$, and $\lfloor K/2 \rfloor$ is the integer part of $K/2$, as well as the rate of 1-bit tensor completion, which is $\asymp d^{-(K-1)/4}$ [8]. To our knowledge, these statistical guarantees

are among the first for binary tensor decomposition, whereas all existing binary tensor decomposition methods have primarily focused on studying the algorithmic convergence properties.

Lastly, we supplement the general statistical properties by proposing a block relaxation algorithm, and establish the corresponding algorithmic convergence properties. Our algorithm-dependent error bound reveals an interesting interplay between the computational efficiency and the statistical convergence. We also illustrate the efficacy of our algorithm through both simulations and real data applications.

2. *The existing work is not cited or compared to (scalable probabilistic tensor factorization for binary and count data, Rai, Hu, Harding and Carin, IJCAI 2015; Boolean Tensor Factorization, Pauli Miettinen, ICDM 2011; Training Binary Weight Networks via Semi-Binary Decomposition, Hu, Li, Wang, Zhang and Cheng, ECCV 2018).*

   Response: Thank you for pointing out those references. The first work, Rai et al. (2015) was already included in the first version of the paper, but we have added a more detailed discussion in the revision. We have also added more relevant papers suggested by you and another referee in **Section 1.2, Related work**.

   In particular, one closely related line of work is higher-order binary tensor decomposition recently studied by [13, 16, 10] (among which [16] is Rai et al. (2015) that you mentioned). We target the same problem. However, our study differs in terms of the optimality properties. In general, there are two types of properties that an estimator possesses. The first type is the algorithm-dependent property that quantifies the impact of a specific algorithm, such as the choice of loss function, initialization, and iterations, on the final estimator. The second type is the statistical property that characterizes the population behavior and is independent of any specific algorithm. Earlier solutions of [13, 16, 10] focused only on the algorithm effectiveness, but did not address the population optimality. By contrast, we study both types of properties in Sections 3 and 4. This allows us to better understand the gap between a specific algorithm and the population optimality, which may in turn offer a useful guide to the algorithm design. This is one of the key differences between our paper and the earlier solutions.

   Another relevant line of work is Boolean tensor decomposition developed by [14, 6, 18, 11] (among which [14, 11] are Miettinen (2011) and Hu et al. (2018) that you mentioned). This is a family of data-driven algorithms that decompose a binary tensor into binary factors. The idea is to use logic operations to replace addition and multiplication in the factorization. These methods also dealt with binary tensor, same as we do, but they took a model-free approach to approximate the data instance. One important difference is that we focus on parameter estimation in a population model. The population interpretation offers useful insight on the effectiveness of dimension reduction. In addition, having a population model allows us to tease apart the algorithmic error versus statistical error. In this respect, our proposal is very different from boolean tensor decomposition. We also numerically compare the two approaches in our newly added simulations. Please see our response below.

   Moreover, we have now added a new **Section 5.3** to numerically compare our method with the existing solutions. Please see our response to your Point 3 below for more details.

3. *The experimental setting is not satisfactory, with a limited setting (simulated data and small datasets), with no comparison to previous work. I would want to see the reconstruction error, compared to other existing methods. The proposed method needs to be compared with binary matrix decomposition.*

Response: We greatly appreciate this suggestion. We have now expanded our numerical experiments considerably.

Numerical comparison: We have now added a new **Section 5.3** to numerically compare our method with the existing solutions. Specifically, we compare with the following alternative solutions for binary tensor factorization (BTF).

- Boolean tensor factorization (BooleanTF) [14, 7, 18]. This method decomposes a binary tensor into binary factors, then cover the binary entries based on a set of logic rules among the factors. We use the implementation of [18].
- Bayesian tensor factorization (BTF_Bayeisan) [17]. This method uses the expectation-maximization approach to decompose a binary tensor into a number of continuous-valued factors. It imposes a Gaussian prior on the factor entries, and a multiplicative gamma process prior on the factor weights $\{\lambda_r\}$.
- Bernoulli tensor factorization with gradient descent (BTF_Gradient) [10]. This method uses a gradient descent algorithm to decompose a binary tensor into continuous-valued factors. We use the implementation in the toolbox of Matlab.

For easy reference, we denote our method by BTF_Alternating, and our implementation can be found at https://github.com/Miaoyanwang/Binary-Tensor. These four methods differ in several ways. BooleanTF is different from the other three in both the cost function and the output format. The rest are all based on the Bernoulli model (2), but with different implementations. BTF_Bayesian employs a Bayesian approach, whereas the other two are frequentist solutions. BTF_Gradient and our method, BTF_Alternating, share the same model, but utilize different optimization algorithms. So the two methods complement each other. On the other hand, in this article, we provide both the algorithm-specific convergence properties, and the algorithm-independent statistical properties including the statistical convergence rate, SNR phase diagram, and mini-max rate. These results are not available in [10] who proposed BTF_Gradient, but these properties hold for BTF_Gradient as well.

We apply the four methods with the default parameters, while selecting the rank R using the recommended approach of each. For our method BTF_Alternating, we use the proposed BIC to select the rank. Since BTF_Gradient does not provide any rank selection criterion, we apply the same R selected by our BIC. For BTF_Alternating, we also set the hyper-parameter α to infinity, which essentially poses no prior on the tensor magnitude. Besides, because BTF_Bayesian only supports the logistic link, we use the logistic link in all three BTF methods.

We evaluate each method by two metrics. The first metric is the root mean square error, $\text{RMSE} = \left(\sqrt{\prod_k d_k}\right)^{-1} \|\widehat{\mathbb{E}(\mathcal{Y})} - \mathbb{E}(\mathcal{Y})\|_F$, where $\widehat{\mathbb{E}(\mathcal{Y})}$ denotes the estimated success probability tensor. For BooleanTF, this quantity is represented as the posterior mean of $\mathcal{Y}$ [14], and for the other three methods, $\widehat{\mathbb{E}(\mathcal{Y})} = \text{logit}(\hat{\Theta})$. The second metic is the misclassification error rate, $\text{MER} = (\prod_k d_k)^{-1} \|\mathbb{1}_{\widehat{\mathbb{E}(\mathcal{Y})} \geq 0.5} - \mathbb{1}_{\mathbb{E}(\mathcal{Y}) \geq 0.5}\|_0$. Here the

indicator function is applied to tensors in an element-wise manner, and $\|\cdot\|_0$ counts the number of non-zero entries in the tensor. These metrics reflect two aspects of the construction error. RMSE summarizes the total estimation error in the success probabilities, whereas MER summarizes the classification errors among 0's and 1's.

We simulate data from two different models, and in both cases, the signal tensors do not necessarily follow an exact low-rank CP structure. Therefore, in addition to method comparison, it also allows us to evaluate the robustness of our method under potential model misspecification. The results are reported in Figure 4 of the paper. For easy reference, we reproduce the plot here in Figure C in the response letter.

The first model is a logic boolean tensor model following the setup in [18]. We first simulate noiseless tensors $\mathcal{Y} = [\![y_{ijk}]\!]$ from the following model,

$$y_{ijk} = \bigvee_{r=1}^{R} \bigwedge_{ijk} a_{ir} b_{jr} c_{kr},\ a_{ir} \sim \text{Ber}(p^a_{ir}),\ b_{jr} \sim \text{Bernoulli}(p^b_{jr}), c_{kr} \sim \text{Bernoulli}(p^c_{kr}),$$

where the binary factor entries $\{a_{ir}\}$, $\{b_{jr}\}$, $\{c_{kr}\}$ are mutually independent with each other, the factor probabilities $\{p^a_{ir}\}$, $\{p^b_{jr}\}$, $\{p^c_{kr}\}$ are generated i.i.d. from Beta(2,4), and $\vee$ and $\wedge$ denote the logical OR and AND operations, respectively. Equivalently, the tensor entry is 1 if and only if there exists one or more components in which all corresponding factor entries are 1. It is easy to verify that

$$\mathbb{E}(y_{ijk}|\{p^a_{ir}, p^b_{jr}, p^c_{kr}\}) = 1 - \prod_{r=1}^{R} \left(1 - p^a_{ir} p^b_{jr} p^c_{kr}\right).$$

We then add contamination bias to $\mathcal{Y}$ by flipping the tensor entries $0 \leftrightarrow 1$ i.i.d. with probability 0.1. We consider the tensor dimension $d_1 = d_2 = d_3 = 50$, and the boolean rank $R \in \{10, 15, 20, 25, 30\}$.

Figure C(a)-(b) shows the performance comparison based on $n_{\text{sim}} = 30$ replications. It is seen that the three BTF methods outperform BooleanTF in RMSE. This shows the advantage of a probabilistic model, upon which all three BTF methods are built. In contrast, BooleanTF seeks patterns in a specific data, but does not address population estimation. For classification, BooleanTF performs reasonably well in distinguishing 0's versus 1's, which agrees with the data mining nature of BoolenTF. It is also interesting to see that MER peaks at $R = 20$. Further investigation reveals that this setting corresponds to the case when the Bernoulli probabilities $\mathbb{E}(\mathcal{Y})$ concentrate around 0.5, which becomes particularly challenging for classification. Actually, the average Bernoulli probability for R =10, 15, 20, 25, 30 is 0.31, 0.44, 0.53, 0.61, 0.68, respectively. Figure C(b) also shows that BTF_Alternating and BTF_Gradient achieve a smaller classification error than BTF_Bayesian. One possible explanation is that the normal prior in BTF_Bayesian has a poor distinguishing power around $\theta \approx 0$, which corresponds to the hardest case when Bernoulli probability ≈ 0.5.

The second model is the stochastic multi-way block model considered in Section 5.2, with the block means $\{c_{m_1 m_2 m_3}\}$ generated from the combinatorial-mean sub-model. Figure C(c)-(d) shows the performance comparison, and a similar pattern is observed. The two frequentist-type BTF methods, BTF_Gradient and BTF_Alternating, behave numerically similarly, and they outperform the other alternatives. In particular, the

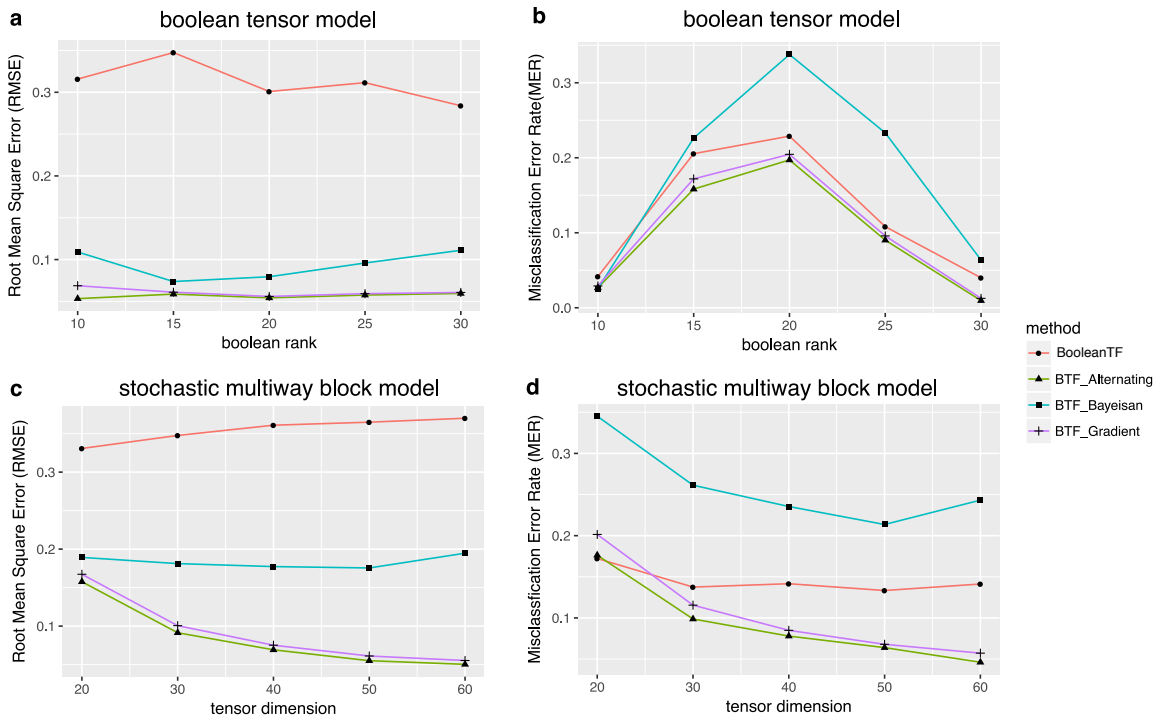

Figure C: Performance comparison in terms of root mean squared error and misclassification error rate. (a)-(b) Estimation errors for the boolean tensor model. (c)-(d) Estimation errors for the stochastic multiway block model.

BTF methods exhibit decaying estimation errors, whereas BooleanTF appears to flatten out as dimension grows. This observation suggests that statistical error is likely more dominating in this setting.

Comparison to binary matrix decomposition: Regarding the relation to binary matrix decomposition, our comparison focuses on the analytical side. First, if we were to apply the matrix version of binary tensor decomposition to a tensor data, by unfolding the tensor into a matrix, the result is suboptimal. Actually, in Section 3.1 we show that, for the scenario when the tensor dimensions are the same in all modes, $d_1 = \ldots = d_K = d$, the convergence rate for estimating Θ using our tensor method is $O(d^{-K+1})$, while the "best" unfolding solution that unfolds a tensor into a near-square matrix [15] gives the convergence rate $O(d^{-\lfloor K/2 \rfloor})$, where $\lfloor K/2 \rfloor$ is the integer part of $K/2$. This gap between the two rates highlights the importance of binary decomposition that specifically takes advantage of the multi-mode structure in tensors. We have added the above comment in **Section 3.2, Page 10, the second to last paragraph**.

Second, in the newly added Section 4.2, we show in Proposition 1(ii) that every local maximizer of $\mathcal{L}$ is an attractor of Algorithm 1. This is a nice property that ensures the exponential decay of the error near a local maximum. Moreover, it reflects some intrinsic difference between tensor decomposition and matrix decomposition, in that the same property often fails for the matrix case. Consider an example of a 2-by-2 matrix. Suppose that the local maximizer is $\Theta^* = \Theta^*(\boldsymbol{e}_1, \boldsymbol{e}_2) = \boldsymbol{e}_1^{\otimes 2} + \boldsymbol{e}_2^{\otimes 2}$, where $\boldsymbol{e}_1, \boldsymbol{e}_2$ are canonical vectors in $\mathbb{R}^2$. There is no region of attraction near the block

variable $\boldsymbol{A}^* = (\boldsymbol{e}_1, \boldsymbol{e}_2)$. In fact, one can construct a point $\boldsymbol{A}^{(0)} = (\boldsymbol{a}_1, \boldsymbol{a}_2)$, with $\boldsymbol{a}_1 = (\sin\theta, \cos\theta)'$, and $\boldsymbol{a}_2 = (\cos\theta, -\sin\theta)'$. The point $\boldsymbol{A}^{(0)}$ can be made arbitrarily close to $\boldsymbol{A}^*$ by tuning θ, but the algorithm iteration initialized from $\boldsymbol{A}^{(0)}$ would never converge to $\boldsymbol{A}^*$. This behavior occurs when the matrix has degenerate singular values. In contract, a 2-by-2-by-2 tensor problem with the maximizer $\tilde{\Theta}^* = \tilde{\Theta}^*(\boldsymbol{e}_1, \boldsymbol{e}_2) = \boldsymbol{e}_1^{\otimes 3} + \boldsymbol{e}_2^{\otimes 3}$ possesses locally unique decomposition, and thus isolated stationary points. The block variable $\boldsymbol{A}^*$ exhibits a basin of attraction with respect to the tensor objective. We have added the above discussion in **Section 4.2, Page 15, the first paragraph after Proposition 1**.

4. *The notation is confusing. In particular, there is confusion between the use of θ, Θ, $f(\theta)$ and $f(\Theta)$.*

   Response: We have now added a clarification in **Section 1.4, Page 5, Lines 6 to 7**, by stating that "we use uppercase letters, such as Θ, $\mathcal{Y}$, $\boldsymbol{A}$, to denote tensors and matrices, and use lowercase letters, such as θ, $\boldsymbol{a}$, to denote scales and vectors."

5. *The organization of the paper needs to be improved. The contributions are not clear. They need to be clearly stated and clearly differentiated from existing work. Existing work should be clearly identified, while currently the novelties are mixed with existing work.*

   Response: We greatly appreciate this suggestion. We have now added a new **Section 1.3, Contributions**, to clarify the main contributions that set our work apart from the existing literature. We have also updated **Section 1.2, Related work**, to offer a more clear and comprehensive literature review.

6. *There is no implementation detail. What software was used? How were parameters chosen? Code?*

   Response: We implement our method using R. We have released our code on GitHub, and have added a reference in **Section 5.3, Page 20, Lines -12 to -11**.

Other comments:

1. *In 1.2: The authors claim that applying continuous-valued tensor decomposition to binary tensor yields an inferior performance. There is no proof of this. Applying a simple threshold can yield good results. In addition the authors mention an issue with flipping the entry coding 0-1. However, this is also an issue for binary methods.*

   Response: Continuous-valued tensor decomposition does not utilize the binary coding in the data. Flipping the entry coding $0 \leftrightarrow 1$ would totally change the decomposition result, and the predicted values for the unobserved entries could fall outside the valid range $[0, 1]$. On the other hand, binary tensor decomposition is invariant to flipping, as reversing the entry coding of $\mathcal{Y}$ only changes the sign, but not the low-rank structure, of the parameter Θ. We have added this clarification in **Section 1.2, Page 2, Lines -4 to -2.**

2. *In 1.2: The authors mention that orthogonality constraints on the factor matrices are too restrictive for tensors. This is not necessarily the case (e.g. Tucker decomposition).*

Response: We have rephrased this statement as: "Such constraints, however, are unnecessary for tensors, since the uniqueness of tensor CP decomposition holds under much milder conditions [4]." Please see **Section 1.2, Page 3, Lines 9 to 10**.

3. *Section 2.2. is hard to read. The link between (3) and (1) needs to be made clearer and expanded. The differences and advantages of the various link functions need to be discussed.*

   Response: We have updated **Section 2.2, Page 6, Lines -13 to -9**. More specifically, we have added the following clarification: "That is, the binary tensor is generated from $\mathcal{Y} = \text{sign}(\Theta + \mathcal{E})$, and the associated latent tensor is $\Theta + \mathcal{E}$. Here the sign function $\text{sign}(x) \stackrel{\text{def}}{=} \mathbb{1}_{\{x \geq 0\}}$ is applied to tensors in an element-wise manner. From the viewpoint of (4), the tensor Θ is not merely an argument in the Bernoulli tensor model (2), it can be interpreted as an underlying, continuous-valued quantity whose noisy discretization gives $\mathcal{Y}$."

   We have also added a paragraph in **Section 2.2, Page 7, the first paragraph after the three examples of the link functions**, discussing various link functions: "The above link functions are common for the Bernoulli model, and the choice is informed by several considerations. The probit is the canonical link based on the Bernoulli likelihood, and has a direct connection with the log-odds of success. The probit is connected to thresholded latent Gaussian tensors. The Laplace has a heavier tail than the normal distribution, so is often more suitable for modeling long-tail data."

4. *3.1: the choice of Loss function is not motivated (e.g. why log, etc).*

   Response: The log-likelihood function is a commonly used loss function in statistics and machine learning; see, e.g., [9, Section 2.6.3, and Section 4.4.1].

5. *5.1: In practice we don't know the rank of the tensor. I would be interested in seeing how the method work when you change the rank of the decomposition, not the rank of the generated tensor. The impact of the link functions is not investigated.*

   Response: Sorry for any confusion, but in all our simulations, we did *not* feed the true rank into our estimation procedure. Instead, we have utilized the BIC criterion we developed in Section 4.3 to select the rank. This criterion aims to balance between the goodness-of-fit for the data and the degree of freedom in the population model. We have added a clarification in **Section 5.1, Page 17, Lines 10 to 11**.

   Regarding the choice of the link function in our simulations, we note that, for the simulation example in Section 5.1, we have employed the logistic link, since it is the most commonly used link function in the generalized linear model literature. In addition, the data has been generated following a logistic link, and we evaluate the finite-sample performance of our method under the correctly specified link function. For the simulation in Section 5.2, we still use the logistic link, but the data has been generated following a probit link. This allows us to evaluate our method under a misspecified link function. For the simulation in Section 5.3, we use the logistic link, because this is the only link function supported by one of the alternative methods to compare to, the Bayesian tensor factorization [17].

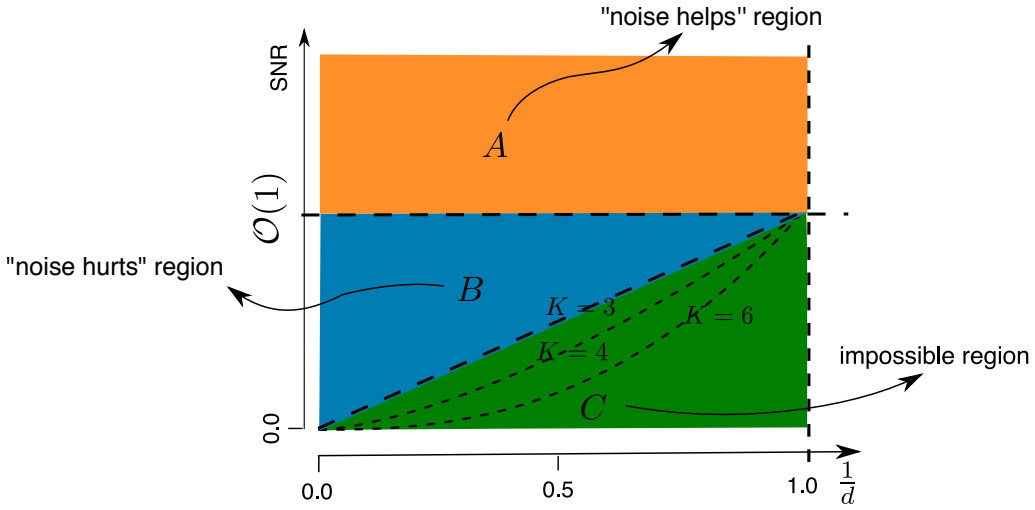

Figure D: Phase diagram with respect to SNR. (A) "Noise helps" region (in yellow): the estimation error decreases with the noise . (B) "Noise hurts" region (in blue): the error increases with the noise. (C) Impossible region (in green): a consistent estimator of Θ is impossible. The dashed line between regions (B) and (C) depicts the boundary $d^{-(K-1)/2}$ as K varies. Note that the origin in the x-axis corresponds to the high-dimensional region, $d^{-(K-1)/2} \to 0$, that is of main interest.

6. *Figure 1.b, with very low noise, I would expect the loss to be lower. How do the authors explain that performance improves when the amount of noise increases? In particular, noise level of log(-3) is inferior to the performance for noise level of log(0.0).*

   Response: This phenomenon seems counter-intuitive, but it is exactly what our theory has described in Section 3.3. More specifically, we characterize the impact of latent signal-to-noise ratio (SNR) on the tensor recovery accuracy, and identify three different phases for tensor recovery according to SNR. In a phase when SNR is sufficiently large, stochastic noise turns out to be helpful, and is in fact essential, for estimating the signal tensor. This effect is related to "dithering" [5] and "perfect separation" [2] phenomenon, and is clearly contrary to the behavior of continuous-valued tensor decomposition.

   We have summarized different phases in Table 1 of the paper, and have also added a new **Figure 1** to graphically illustrate the three phrases under the case when $d_1 = \ldots = d_K = d$. For easy reference, we produce the plot here in Figure D. The detailed explanation is given in **Section 3.3, Page 11, the last paragraph**.

7. *In 5.2 the authors mention that the signal tensor does not always follow the CP structure. What does this mean? All tensors admit a CP decomposition. The block structure is equally unconvincing.*

   Response: You are right that all tensors admit a CP structure. What we meant is a CP with an explicit *low-rank* structure. We have rephrased the original statement in **Section 5.2, Page 18, Lines -7 to -6**, by saying that "Under this model, the signal tensor does not necessarily admit a low-rank structure."

   Regarding the block structure, we have added the rationale in **Section 5.2, Page 18, Lines -10 to -7**, by stating that "We next evaluate our method under the stochastic

Dataset	Non-zeros	Tensor decomposition method			
		Binary (logistic link)		Continuous-valued	
		AUC	RMSE	AUC	RMSE
Kinship	3.80%	0.9708	1.2×10^{-4}	0.9436	1.4×10^{-3}
Nations	21.1%	0.9169	1.1×10^{-2}	0.8619	2.2×10^{-2}
Enron	0.01%	0.9432	6.4×10^{-3}	0.7956	6.3×10^{-5}
HCP	35.3%	0.9860	1.3×10^{-3}	0.9314	1.4×10^{-2}

Table B: Tensor completion for the four real-world binary tensor datasets. Two methods are compared: the proposed binary tensor decomposition, and the classical continuous-valued tensor decomposition.

multi-way block model, which can be viewed as a higher-order generalization of the stochastic block model that is commonly used for random graphs, network analysis, and community detection [3, 1]."

8. *In 6, the real-world datasets are very small, and there is no comparison to existing work. What about comparing to real-valued decomposition + rounding even?*

   Response: We have considered four real-world datasets, among which the *Enron* dataset consists of more than 3 million ($581 \times 124 \times 48 = 3,458,112$) entries and the *Kinship* dataset consists of 281,216 ($= 104 \times 104 \times 26$) entries. The *Kinship*, *Nations*, and *Enron* datasets are commonly used in the tensor literature. These datasets appear to be fairly representative to serve our purpose.

   In the first task of binary tensor completion, we have indeed compared our binary tensor decomposition method with the real-valued tensor decomposition method. Please see **Section 6, Page 23, the last paragraph, and Page 24, the first paragraph**. Specifically, we report in Table 4 of the paper the area under the ROC curve (AUC) and RMSE, averaged over five random splits of the training and testing data. It is clearly seen that the binary tensor decomposition substantially outperforms the classical continuous-valued tensor decomposition. In all datasets, the former obtains a much higher AUC and mostly a lower RMSE. We also report in Table 4 the percentage of nonzero entries for each data. It is seen that our decomposition method performs well even in the sparse setting. For instance, for the Enron dataset, only 0.01% of the entries are non-zero. The classical decomposition almost blindly assigns 0 to all the hold-out testing entires, resulting in a poor AUC of 79.6%. By comparison, our binary tensor decomposition achieves a much higher classification accuracy, with AUC = 94.3%. For easy reference, we reproduce those results of Table 4 in Table B here in the response letter. We did not compare with alternative methods in the second task of clustering, as we do not know the true clustering membership in the real data.

References

[1] Emmanuel Abbe. Community detection and stochastic block models: recent developments. *The Journal of Machine Learning Research*, 18(1):6446–6531, 2017.

[2] Adelin Albert and John A Anderson. On the existence of maximum likelihood estimates in logistic regression models. *Biometrika*, 71(1):1–10, 1984.

[3] 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.

[4] Aditya Bhaskara, Moses Charikar, and Aravindan Vijayaraghavan. Uniqueness of tensor decompositions with applications to polynomial identifiability. In *Conference on Learning Theory*, pages 742–778, 2014.

[5] Mark A Davenport, Yaniv Plan, Ewout Van Den Berg, and Mary Wootters. 1-bit matrix completion. *Information and Inference: A Journal of the IMA*, 3(3):189–223, 2014.

[6] Dóra Erdos and Pauli Miettinen. Discovering facts with boolean tensor tucker decomposition. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 1569–1572. ACM, 2013.

[7] Dóra Erdos and Pauli Miettinen. Walk'n'merge: a scalable algorithm for boolean tensor factorization. In *2013 IEEE 13th International Conference on Data Mining*, pages 1037–1042. IEEE, 2013.

[8] Navid Ghadermarzy, Yaniv Plan, and Ozgur Yilmaz. Learning tensors from partial binary measurements. *arXiv preprint arXiv:1804.00108*, 2018.

[9] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 edition, 2009.

[10] David Hong, Tamara G Kolda, and Jed A Duersch. Generalized canonical polyadic tensor decomposition. *arXiv preprint arXiv:1808.07452*, 2018.

[11] Qinghao Hu, Gang Li, Peisong Wang, Yifan Zhang, and Jian Cheng. Training binary weight networks via semi-binary decomposition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 637–653, 2018.

[12] Joseph B Kruskal. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear algebra and its applications*, 18(2):95–138, 1977.

[13] Jakub Mažgut, Peter Tiňo, Mikael Bodén, and Hong Yan. Dimensionality reduction and topographic mapping of binary tensors. *Pattern Analysis and Applications*, 17(3):497–515, 2014.

[14] Pauli Miettinen. Boolean tensor factorizations. In *2011 IEEE 11th International Conference on Data Mining*, pages 447–456. IEEE, 2011.

[15] Cun Mu, Bo Huang, John Wright, and Donald Goldfarb. Square deal: Lower bounds and improved relaxations for tensor recovery. In *International Conference on Machine Learning*, pages 73–81, 2014.

[16] Piyush Rai, Changwei Hu, Matthew Harding, and Lawrence Carin. Scalable probabilistic tensor factorization for binary and count data. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, pages 3770–3776, 2015.

[17] Piyush Rai, Yingjian Wang, Shengbo Guo, Gary Chen, David Dunson, and Lawrence Carin. Scalable bayesian low-rank decomposition of incomplete multiway tensors. In *International Conference on Machine Learning*, pages 1800–1808, 2014.

[18] Tammo Rukat, Chris Holmes, and Christopher Yau. Probabilistic boolean tensor decomposition. In *International conference on machine learning*, pages 4410–4419, 2018.

[19] Will Wei Sun, Junwei Lu, Han Liu, and Guang Cheng. Provable sparse tensor decomposition. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(3):899–916, 2017.

[20] André Uschmajew. Local convergence of the alternating least squares algorithm for canonical tensor approximation. *SIAM Journal on Matrix Analysis and Applications*, 33(2):639–652, 2012.

[21] Hua Zhou, Lexin Li, and Hongtu Zhu. Tensor regression with applications in neuroimaging data analysis. *Journal of the American Statistical Association*, 108(502):540–552, 2013.