Tensor Train Decomposition for BN2A Structure Learning

Iván Pérez CABRERA@UTIA.CAS.CZ
Institute of Information Theory and Automation, Czech Academy of Sciences
Institute of Computer Science, Czech Academy of Sciences
Faculty of Mathematics and Physics, Charles University
Jiří Vomlel VOMLEL@UTIA.CAS.CZ
Institute of Information Theory and Automation, Czech Academy of Sciences
Patrícia Martinková MARTINKOVA@CS.CAS.CZ
Institute of Computer Science, Czech Academy of Sciences

Editors: Gustau Camps-Valls, Manuele Leonelli and Gherardo Varando

Abstract

BN2A models are bipartite Bayesian networks with binary hidden and observed variables, characterized by Noisy-AND conditional probability tables with leak and penalty parameters. A fundamental challenge in BN2A modeling is identifying the bipartite graph structure, the Q-matrix in cognitive diagnostic models, from observed data alone. Existing tensor-based identification methods face the curse of dimensionality: the joint probability tensor has 2^J entries for J observed variables, and constructive algorithms such as that of Gu (2026) require materializing it explicitly. We bridge BN2A structure learning and tensor network methods by showing that BN2A probability tensors have Tensor Train (TT) rank at most 2^K, where K is the number of hidden variables, reducing storage from $O(2^J)$ to $O(J \cdot 4^K)$, a bound that follows from a known equivalence between BN2A and Restricted Boltzmann Machines. Building on this representation, we show that Gu's rank tests can be performed directly on TT cores via core contractions, without ever materializing the full tensor. Empirical validation on simulated data with $J = 15$ and $K = 3$ confirms the rank bound and achieves perfect structure learning; we further characterize the sample-size regime, showing that recovery is robust for n as small as 30% of 2^J. We also report two honest explorations of the boundaries of the approach: greedy mode ordering reduces the TT parameter count of the population tensor by 13% and, once an order-sensitive ambiguity in pure-item detection is resolved, composes fully with Gu's rank tests, although the storage benefit vanishes on noisy empirical tensors; and Gu's Step 2 itself degrades as J grows with K fixed, a limitation of the underlying algorithm rather than of the TT representation.

Keywords: BN2A models, Bipartite Bayesian networks, Structure learning, Tensor Train decomposition, Restricted Boltzmann Machines, Cognitive diagnostic models

1. Introduction

Bayesian networks with hidden variables are widely used to model latent structure in observed data. A particular subclass is the family of *bipartite* Bayesian networks with binary hidden and observed variables, in which each observed variable depends only on a subset of the hidden variables and the conditional probability tables (CPTs) take a canonical form (Díez and Druzdzel, 2006). The BN2A model (Pérez and Vomlel, 2024), a Bayesian network formulation equivalent to the Reduced Reparametrized Unified Model (RRUM)

© 2026 I. Pérez, J. Vomlel & P. Martinková.

from psychometrics (Hartz and Roussos, 2008), assumes a Noisy-AND mechanism: the probability that an observed variable Y_j takes the value 1 equals the *leak* parameter (close to 1) when all required hidden variables are active, and is reduced by a multiplicative penalty in $(0, 1)$ for each required hidden variable X_k that is inactive. The bipartite graph encoding which hidden variables are parents of which observed variables is summarized by a binary matrix called the Q-matrix.

Recovering the bipartite structure from observed data alone, the structure learning problem for BN2A, is a long-standing challenge. Gu (2026) recently proposed a fully constructive algorithm based on rank tests of unfoldings of the joint probability tensor $\mathcal{T}$. The approach is principled, but it requires materializing $\mathcal{T}$ explicitly, and with 2^J entries for J binary observed variables this becomes impractical as J grows.

Tensor decompositions have been used in PGMs to accelerate inference and to estimate parameters under a known structure (Section 2.2); to our knowledge, they have not been applied to the inverse problem of learning the structure itself.

This paper bridges these two lines of work through four contributions. First, we show that the joint probability tensor of a BN2A model admits a Tensor Train representation with TT-rank bounded by 2^K, an immediate consequence of the equivalence between BN2A and binary Restricted Boltzmann Machines (Chen et al., 2018), reducing storage from $O(2^J)$ to $O(J \cdot 4^K)$ (Section 3). Second, we show that the rank tests of Gu (2026) can be performed directly on TT cores via core contractions, without ever materializing $\mathcal{T}$ (Section 4). Third, we validate the pipeline experimentally on BN2A models with $J = 15$ and $K = 3$, learning the structure exactly on the empirical tensor and characterizing the sample-size regime in which it is robust (Section 4). Finally, we report two honest explorations of the boundaries of the approach: the greedy mode-ordering algorithm of Tichavský and Straka (2025) reduces TT model size on the population tensor and composes with Gu's rank tests once an order-sensitive ambiguity in pure-item detection is resolved, though without net storage gain on empirical tensors; and Step 2 of Gu (2026)'s algorithm degrades as J grows with K fixed, a limitation inherited by any pipeline built on those tests (Section 5).

The rest of the paper is organized as follows. Section 2 sets up notation and positions the work in the literature. Sections 3 and 4 present the TT representation of BN2A and the TT-based implementation of Gu's algorithm, respectively. Section 5 discusses the two boundary explorations and outlines future directions.

2. Background

This section sets up the problem and positions our work relative to prior tensor-based approaches in probabilistic graphical models (PGMs).

2.1. BN2A models and the structure learning problem

A BN2A model is a bipartite Bayesian network with K binary hidden variables $\mathbf{X} = (X_1, \ldots, X_K) \in \{0, 1\}^K$ and J binary observed variables $\mathbf{Y} = (Y_1, \ldots, Y_J) \in \{0, 1\}^J$ (Figure 1). We adopt the terminology of cognitive diagnostic modeling (CDM) where convenient: hidden variables are referred to as *attributes* and observed variables as *items*, with the bipartite graph encoded by a binary $J \times K$ matrix called the Q-matrix. This convention

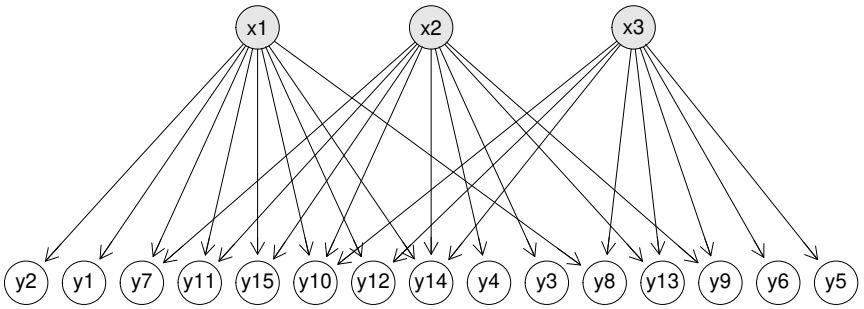


Figure 1: Structure of a BN2A model. Top layer: K hidden binary variables. Bottom layer: J observed binary variables. This particular graph corresponds to the $J = 15$, $K = 3$ configuration used in our experiments (Section 4): six pure items (each connected to a single attribute), seven pair items, and two triple items.

aligns with the algorithm of Gu (2026) that we build upon and adapt to a Tensor Train representation in subsequent sections.

In BN2A models, the CPT $P(Y_j \mid \mathbf{X}_{pa(Y_j)})$ takes the *Leak Noisy-AND* form

$$P(Y_j = 1 \mid \mathbf{X}_{pa(Y_j)} = \mathbf{x}_{pa(Y_j)}) = q_{j,0} \prod_{i \in pa(Y_j)} (q_{j,i})^{1-x_i}, \tag{1}$$

where the parent set $pa(Y_j) \subseteq \{1, \ldots, K\}$ encodes the bipartite graph structure. The scalar $q_{j,0} \in (0, 1]$ is the *leak probability*: the probability that Y_j is active when all its parents $pa(Y_j)$ are active. The scalars $q_{j,i} \in (0, 1]$, for $i \in pa(Y_j)$, are the *penalty parameters*: multiplicative factors that reduce the activation probability of Y_j when the parent X_i is inactive. The model is monotone: activating additional parents can only increase the activation probability of Y_j.

Recovering the Q-matrix from observed data alone is referred to as the *Q-matrix identification problem* (Gu, 2026); in PGM terminology, this is the *structure learning* problem for the bipartite BN2A graph: identifying which edges $X_k \to Y_j$ are present.

Gu (2026) proposed a constructive algorithm for BN2A structure learning based on rank tests of unfoldings of the joint probability tensor of the observed responses,

$$\mathcal{T}[y_1, \ldots, y_J] = P(Y_1 = y_1, \ldots, Y_J = y_J). \tag{2}$$

The algorithm proceeds in two steps. **Step 1** identifies items that load on a single attribute (*pure items*) by detecting pairs (j_1, j_2) for which the matrix unfolding $\mathcal{T}_{\{j_1,j_2\} \mid [J]\setminus\{j_1,j_2\}}$ has rank at most 2. **Step 2** classifies multi-attribute items by examining higher-order unfoldings with rank threshold 2^{K-1}.

The bottleneck is the size of $\mathcal{T}$, whose number of entries, 2^J, scales exponentially in J and becomes impractical for moderately large item sets. Already for $J = 15$ the tensor has 32,768 entries; for $J = 20$ around 10^6; and the SVDs required by the rank tests grow correspondingly.

2.2. Tensor decompositions in PGMs

Tensor methods have been proposed to mitigate exponential blow-up in PGMs along three main directions: inference, parameter estimation, and structure learning.

CP rank-one decomposition for inference. Savicky and Vomlel (2007) introduced CP rank-one decomposition of canonical CPTs (Noisy-OR, Noisy-MAX, Noisy-AND, Noisy-XOR) for efficient probabilistic inference. Vomlel and Tichavský (2014) extended this analysis to the broader class of Noisy-threshold models, which includes both Noisy-OR and Noisy-AND as special cases. They establish that the symmetric tensor rank of the ℓ-out-of-k threshold tensor in the complex domain equals $\max\{\ell+1, k-\ell+1\}$, yielding a polynomial-size junction tree where standard moralization yields exponential cliques. Their experiments on Noisy-threshold variants of QMR-DT (the decision-theoretic reformulation of the Quick Medical Reference diagnostic network, itself a large bipartite two-level Bayesian network (Shwe et al., 1991)) show inference-time gains of several orders of magnitude over the parent-divorcing method.

Moment-based tensor methods for latent variable models. A complementary line of work uses tensor decompositions for *parameter estimation* in latent variable models. Anandkumar et al. (2014) show that the low-order observable moments (typically second and third order) of a wide class of models, including Gaussian mixtures, hidden Markov models, and latent Dirichlet allocation, exhibit a symmetric tensor structure from which the model parameters can be recovered via an orthogonal tensor decomposition, computable by power iterations with provable guarantees. That approach operates on low-order moments and estimates parameters for a fixed model class with a known number of latent components; the problem we address is different in both respects, as the rank tests of Gu (2026) interrogate unfoldings of the full joint tensor to recover the *structure* itself.

Tensor Train for inference. Ducamp et al. (2020) apply the Tensor Train (TT) format to message-passing in junction trees, redefining clique-separator products and marginalizations directly in compressed form. They report substantial gains for large dynamic Bayesian networks and for networks such as `Munin1` and `Link` where the standard Shafer-Shenoy algorithm runs out of memory. Beyond inference, Chen et al. (2018) characterize the expressive power of tensor networks for latent variable models, establishing that binary Restricted Boltzmann Machines with K hidden units admit TT representations with rank at most 2^K; this equivalence is the foundation of our rank bound in Section 3.

Tensor Train for structure learning. The works above use tensor decompositions either for *inference* (Savicky and Vomlel, 2007; Vomlel and Tichavský, 2014; Ducamp et al., 2020; the graph structure is assumed known and the CPTs associated with it are compressed) or for *parameter estimation* under a known structure (Anandkumar et al., 2014). Our problem is the inverse: we are given observed data and seek to *recover* the bipartite graph structure. To the best of our knowledge, the TT format has not previously been applied to structure learning in PGMs. In this work we build on the full-tensor algorithm of Gu (2026) and evaluate the TT-based variant experimentally. Table 1 summarizes the positioning of the methods discussed above.

Table 1: Storage complexity of tensor representations used in tensor-based PGM methods. J: number of observed variables; K: number of hidden variables; R: TT-rank.

Method	Model class	Complexity	Objective
Full CPT (baseline)	BN2A	$O(2^J)$	Structure learning
CP rank-one	Noisy-threshold BNs	$O(J^2)$	Inference
Moment CP	Latent variable models	$O(J^3)$	Parameter estimation
TT for CPTs	General BNs	$O(J \cdot R^2)$	Inference
TT + Gu rank tests	BN2A	$O(J \cdot 4^K)$	**Structure learning**

Relation to constraint-based and score-based structure learning. The algorithm of Gu (2026) is neither constraint-based nor score-based: it searches no space of candidate structures and optimizes no score, but instead reads the structure off rank conditions on the joint distribution. Rank conditions of this kind play, for latent structure, a role analogous to that of conditional-independence constraints in constraint-based methods, and remain informative after the hidden variables are marginalized out (Allman et al., 2009). If the hidden variables were observed, structure recovery would decompose into J parent-set identification problems with cost governed by the 2^K conditional probability table entries per item; the TT-based pipeline retains this type of scaling (linear in J, exponential only in K) even though they are latent.

2.3. Tensor Train decomposition

The Tensor Train (TT) format (Oseledets, 2011) represents a tensor $\mathcal{T} \in \mathbb{R}^{n_1 \times \cdots \times n_J}$ as a sequence of order-three *cores* $\mathcal{G}_j \in \mathbb{R}^{r_{j-1} \times n_j \times r_j}$, $j = 1, \ldots, J$, with $r_0 = r_J = 1$, such that for every index tuple,

$$\mathcal{T}[i_1, \ldots, i_J] = \mathcal{G}_1[:, i_1, :]\, \mathcal{G}_2[:, i_2, :] \cdots \mathcal{G}_J[:, i_J, :]. \tag{3}$$

The integers r_j are the *TT-ranks* (or *bond dimensions*) and govern model complexity. The total storage is $\sum_{j=1}^{J} r_{j-1} n_j r_j$, which is linear in J when the ranks are bounded.

Equation (3) is conveniently visualized as a *tensor network diagram* (Figure 2): each tensor is drawn as a node, each index as a leg, and an edge joining two nodes denotes summation over the shared index. In this language the joint tensor is a single node with J legs, its TT representation is a chain of order-three cores, and every operation in Section 4 is a contraction of this chain.

A canonical algorithm for constructing a TT representation from a fully-specified tensor is the TT-SVD (Oseledets, 2011), which proceeds by sequential SVDs of unfoldings. In our setting all observed variables are binary, so $n_j = 2$ for every j.

3. Tensor Train Representation of BN2A

This section establishes the central observation that enables our approach: the joint probability tensor $\mathcal{T}$ of a BN2A model admits a Tensor Train representation with bond dimension

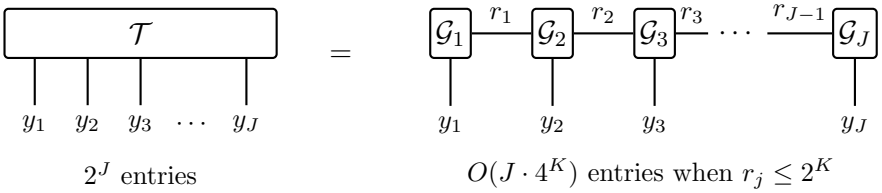


Figure 2: The joint probability tensor of a BN2A model (left) and its Tensor Train representation (right) as tensor network diagrams: nodes are tensors, dangling legs are indices, and an edge joining two nodes denotes summation over the shared index. For BN2A models the bond dimensions satisfy $r_j \leq 2^K$ (Section 3).

at most 2^K, independent of J. We obtain this bound via the equivalence between BN2A and Restricted Boltzmann Machines (RBMs), and validate it numerically across a range of configurations.

3.1. From BN2A to a marginalized tensor with low TT-rank

The joint tensor $\mathcal{T}$ in Equation (2) can be written by marginalizing over the hidden pattern $\mathbf{x}$,

$$\mathcal{T}[y_1, \ldots, y_J] = \sum_{\mathbf{x} \in \{0,1\}^K} \pi(\mathbf{x}) \prod_{j=1}^{J} P(Y_j = y_j \mid \mathbf{x}), \tag{4}$$

where $\pi(\mathbf{x})$ is the prior over hidden patterns and the conditional probabilities are given by Equation (1). The sum has 2^K terms; given the bipartite structure, this factorization matches exactly the marginalization structure of a Restricted Boltzmann Machine over binary visible and hidden units.

Chen et al. (2018) establish that a binary RBM with K hidden units and J visible units admits a Tensor Train representation of the joint distribution over visible units with TT-rank at most 2^K. The bound has a simple combinatorial interpretation: the bond dimension at each cut needs only to carry the hidden state across the chain, and the hidden state space has cardinality 2^K. Translating to BN2A, every BN2A model induces a TT representation of $\mathcal{T}$ with bond dimensions bounded by 2^K.

Proposition 1 (TT-rank bound for BN2A) *Let $\mathcal{T} \in \mathbb{R}^{2 \times \cdots \times 2}$ be the joint probability tensor of a BN2A model with J observed variables and K hidden variables, as defined in Equation (2). Then $\mathcal{T}$ admits an exact TT representation*

$$\mathcal{T}[y_1, \ldots, y_J] = \mathcal{G}_1[:, y_1, :]\, \mathcal{G}_2[:, y_2, :] \cdots \mathcal{G}_J[:, y_J, :]$$

with TT-ranks $r_j \leq 2^K$ for all $j = 1, \ldots, J-1$.

The proposition follows directly from Chen et al. (2018, Theorem 1) applied to the BN2A factorization in Equation (4): BN2A is structurally a binary RBM, and the TT cores can be read off the hidden-state contractions. We refer the reader to Chen et al. (2018) for the explicit construction.

Table 2: Numerical validation of the TT-rank bound for the population tensor $\mathcal{T}$ of BN2A models. The observed maximum bond dimension matches the theoretical bound 2^K in every configuration (4 for $K = 2$, 8 for $K = 3$).

		$K = 2$		$K = 3$	
J	$\lvert\mathcal{T}\rvert$	TT params	Compression	TT params	Compression
10	1,024	80	$\sim 13\times$	~ 480	$\sim 2\times$
15	32,768	120	$\sim 270\times$	$\sim 1{,}100$	$\sim 30\times$
20	1,048,576	160	$\sim 6{,}550\times$	$\sim 1{,}900$	$\sim 550\times$

Remark 2 *The bound 2^K is a uniform upper bound across all bonds; in practice the observed ranks vary along the chain, typically saturating near the middle and decreasing toward both ends. The bound depends only on K, not on J: increasing J does not increase the maximum bond dimension. This is the key structural fact that enables scalability.*

3.2. Storage complexity

The storage of the TT representation is $\sum_{j=1}^{J} 2 \cdot r_{j-1} \cdot r_j$ (each core $\mathcal{G}_j$ has size $r_{j-1} \times 2 \times r_j$). With ranks bounded by $r_j \le 2^K$ for $1 \le j \le J-1$ and $r_0 = r_J = 1$, this is at most

$$\underbrace{2 \cdot 1 \cdot 2^K}_{j=1} + \underbrace{(J-2) \cdot 2 \cdot 4^K}_{2 \le j \le J-1} + \underbrace{2 \cdot 2^K \cdot 1}_{j=J} = 4 \cdot 2^K + (J-2) \cdot 2 \cdot 4^K = O(J \cdot 4^K).$$

This is linear in J for fixed K, in contrast with the 2^J entries of the full tensor. For instance, with $J = 20$ and $K = 3$, the full tensor has $\sim 10^6$ entries while the TT representation has at most $\sim 2{,}500$ parameters.

3.3. Numerical validation

We construct $\mathcal{T}$ exactly via Equation (4) for a range of (J, K) configurations and apply TT-SVD (Oseledets, 2011) with truncation tolerance 10^{-12} relative to the largest singular value at each bond. For each configuration, a Q-matrix $\mathbf{Q}$ is chosen to include rows with a single 1 and rows with multiple 1s; leak parameters $q_{j,0}$ and penalty parameters $q_{j,k}$ are sampled from $\{0.70, 0.75, \ldots, 0.95\}$ and $\{0.10, 0.20, 0.30, 0.40\}$ respectively, ensuring nontrivial structure with realistic values. Table 2 reports the observed maximum bond dimension and the resulting compression factor.

Two observations from Table 2 are worth noting. First, the observed maximum TT-rank matches the theoretical bound 2^K exactly in every configuration: the bound is tight on the population tensor for nontrivial Q-matrix $\mathbf{Q}$. Second, the compression factor grows rapidly with J for fixed K, consistent with the $O(J \cdot 4^K)/2^J$ scaling. For $J = 20$, $K = 2$, the TT representation is over three orders of magnitude more compact than the full tensor; for $J = 20$, $K = 3$, more than two orders of magnitude.

Bond profile. The maximum TT-rank reported in Table 2 is achieved only at the central bonds; the bonds near both ends of the chain are typically smaller. For instance, for $J = 15$, $K = 3$ on the population tensor, the observed bond profile is $(1, 2, 2, 4, 4, 8, 8, 8, 8, 8, 8, 8, 8, 4, 2, 1)$: eight bonds saturate at the bound $2^K = 8$, and the bonds shrink toward both endpoints of the chain. This pattern reflects the structural fact that the leftmost (resp. rightmost) cores only need to encode partial information about the hidden state $\mathbf{x}$; the full 2^K configurations are only required once enough items have been incorporated.

Empirical tensor versus population tensor. The bound in Proposition 1 applies to the *population* tensor $\mathcal{T}$, i.e., the true joint distribution. In practice, structure learning operates on the *empirical* tensor $\widehat{\mathcal{T}}$ obtained from a sample of n observed response vectors. Sampling noise inflates the effective rank under any fixed truncation tolerance, since spurious small singular values may exceed the threshold. In Section 4 we address this by enforcing the theoretical cap $r_j \leq 2^K$ during TT-SVD, which yields a numerically stable approximation that retains the structural rank information needed by the rank tests of Gu (2026).

Implementation. Computing the TT-SVD for the largest configuration in Table 2 ($J = 20$, $K = 3$) takes approximately 0.2 seconds; all other configurations complete in well under a second (hardware and code availability in Section 4.4).

4. Structure Learning on TT Cores

The structure learning algorithm of Gu (2026) operates on rank tests of matrix unfoldings of the joint tensor $\mathcal{T}$. In its original form, the algorithm materializes $\mathcal{T}$ explicitly, whose number of entries, 2^J, scales exponentially in J and becomes impractical for moderately large sets of observed variables. We show that the same rank tests can be performed on a TT representation of $\mathcal{T}$, by evaluating the singular values of the relevant unfoldings directly from the TT cores, without reconstructing the full tensor or forming any object whose size grows exponentially in J. Combined with the storage savings of Section 3, this yields a structure learning pipeline whose memory footprint is $O(J \cdot 4^K)$ rather than $O(2^J)$.

4.1. Unfoldings via core contractions

Fix a subset $S \subseteq \{1, \ldots, J\}$ and let $\bar{S} = \{1, \ldots, J\} \setminus S$. The matrix unfolding $\mathcal{T}_{S|\bar{S}}$ has rows indexed by binary assignments $\mathbf{y}_S \in \{0, 1\}^{|S|}$ to the variables in S and columns indexed by binary assignments $\mathbf{y}_{\bar{S}} \in \{0, 1\}^{|\bar{S}|}$ to the remaining variables. The rank tests of Gu (2026) examine the rank of $\mathcal{T}_{S|\bar{S}}$ for specific choices of S that we discuss in Section 4.2.

Given a TT representation $\mathcal{T}[y_1, \ldots, y_J] = \mathcal{G}_1[:, y_1, :] \cdots \mathcal{G}_J[:, y_J, :]$, each entry of the unfolding is a product of indexed core slices,

$$\mathcal{T}_{S|\bar{S}}[\mathbf{y}_S, \mathbf{y}_{\bar{S}}] \;=\; \prod_{j=1}^{J} \mathcal{H}_j(y_j), \qquad \mathcal{H}_j(y_j) = \mathcal{G}_j[:, y_j, :] \in \mathbb{R}^{r_{j-1} \times r_j}, \tag{5}$$

where the value of y_j is read from $\mathbf{y}_S$ if $j \in S$ and from $\mathbf{y}_{\bar{S}}$ if $j \in \bar{S}$, and $r_0 = r_J = 1$ collapses the product to a scalar. Assembling the unfolding entry by entry, however, produces a matrix with 2^J entries, as many as $\mathcal{T}$ itself.

The rank tests require only the singular values of $\mathcal{T}_{S|\bar{S}}$, and these can be obtained without forming the unfolding at all. Let $A = \mathcal{T}_{S|\bar{S}}\mathcal{T}_{S|\bar{S}}^{\top} \in \mathbb{R}^{2^{|S|}\times 2^{|S|}}$ be the Gram matrix of the unfolding; its eigenvalues are the squared singular values of $\mathcal{T}_{S|\bar{S}}$. Writing both copies of $\mathcal{T}$ in TT form and exchanging the sum over $\mathbf{y}_{\bar{S}}$ with the products along the chain yields the transfer recursion

$$A[\mathbf{y}_S, \mathbf{y}'_S] \;=\; V_J, \qquad V_j \;=\; \begin{cases} \mathcal{H}_j(y_j)^{\top}\, V_{j-1}\, \mathcal{H}_j(y'_j) & \text{if } j \in S, \\ \sum_{y\in\{0,1\}} \mathcal{H}_j(y)^{\top}\, V_{j-1}\, \mathcal{H}_j(y) & \text{if } j \in \bar{S}, \end{cases} \tag{6}$$

with $V_0 = 1$, where y_j and y'_j denote the entries of $\mathbf{y}_S$ and $\mathbf{y}'_S$ at position j. Every intermediate V_j is an $r_j \times r_j$ matrix with $r_j \le 2^K$, so no object of size exponential in J is ever formed: one Gram entry costs $O(J{\cdot}8^K)$ time and $O(4^K)$ memory, and the Gram matrix itself has $2^{|S|} \times 2^{|S|}$ entries with $|S| \le K+1$ in all tests we perform. Forming the Gram matrix squares the condition number of the problem, which halves the attainable precision on the smallest singular values; this is immaterial for our purposes, since the singular-value ratios being tested lie many orders of magnitude above machine precision.

4.2. Gu's rank tests on TT cores

The algorithm of Gu (2026) proceeds in two steps, each consisting of rank tests on specific unfoldings.

Step 1: pure items. For every pair (j_1, j_2) with $j_1 < j_2$, the algorithm computes the unfolding $\mathcal{T}_{\{j_1,j_2\}|\bar{S}}$ where $\bar{S} = \{1,\dots,J\} \setminus \{j_1, j_2\}$, and tests whether its rank is at most 2. Pairs that pass the test are flagged as candidate *pure pairs*; the connected components of the resulting graph identify groups of items associated with the same hidden variable.

In our pipeline this test is implemented by computing the 4×4 Gram matrix of the unfolding via Equation (6) with $S = \{j_1, j_2\}$ and taking its eigenvalues $\lambda_1 \ge \cdots \ge \lambda_4$; the singular values are $\sigma_i = \sqrt{\lambda_i}$, and the numerical rank is determined by comparing σ_3/σ_1 against a threshold τ_1. We discuss the choice of τ_1 in Section 4.4.

Step 2: multi-attribute items. For each item j that was not flagged as pure in Step 1, the algorithm computes a sequence of higher-order unfoldings of the form $\mathcal{T}_{S_1|\bar{S}_1}$ where S_1 contains j together with one representative pure item per hidden variable, and tests whether the rank is at most 2^{K-1}. The membership pattern of j in the rank-deficient configurations encodes its corresponding row in the Q-matrix. We refer the reader to Gu (2026) for the precise specification of the unfoldings and the decoding rule; in our pipeline each such test is implemented via Equation (6) with the corresponding $S = S_1$.

4.3. Algorithm

We summarize the full pipeline in Algorithm 1. The input is a TT representation of $\widehat{\mathcal{T}}$, obtained by applying TT-SVD with rank cap 2^K to the empirical tensor. The output is the recovered Q-matrix $\widehat{\mathbf{Q}}$.

Algorithm 1 performs $\binom{J}{2} = O(J^2)$ rank tests in Step 1 and a number of additional rank tests in Step 2 determined by Proposition 2 of Gu (2026). Each test computes the Gram matrix of one unfolding $\mathcal{T}_{S|\bar{S}}$ via Equation (6) (with $|S| = 2$ in Step 1 and $|S| \le K+1$ in

Algorithm 1: Structure learning of BN2A via TT cores

Input: Empirical tensor $\widehat{\mathcal{T}}$; hidden dimension K; thresholds τ_1, τ_2
Output: Recovered Q-matrix $\widehat{\mathbf{Q}}$

Compute TT cores $\{\mathcal{G}_j\}_{j=1}^{J}$ from $\widehat{\mathcal{T}}$ via TT-SVD with rank cap $r_{\max} = 2^K$

`/* Step 1: pure-pair detection */`
for *each pair* (j_1, j_2) *with* $j_1 < j_2$ **do**
  Compute Gram matrix $A \leftarrow \mathcal{T}_{\{j_1,j_2\}|\bar{S}} \mathcal{T}_{\{j_1,j_2\}|\bar{S}}^{\top}$ via Eq. (6)
  Compute eigenvalues $\lambda_1 \geq \cdots \geq \lambda_4$ of A and set $\sigma_i = \sqrt{\lambda_i}$
  if $\sigma_3/\sigma_1 < \tau_1$ **then** mark (j_1, j_2) as candidate pure pair
end
Cluster items by connected components of the candidate-pure-pair graph

`/* Step 2: multi-attribute classification */`
for *each item* j *not in any pure cluster* **do**
  for *each subset* S *specified by Proposition 2 of Gu (2026) involving* j **do**
    Compute Gram matrix $A \leftarrow \mathcal{T}_{S|\bar{S}} \mathcal{T}_{S|\bar{S}}^{\top}$ via Eq. (6)
    Compute $\sigma_i = \sqrt{\lambda_i}$ from the eigenvalues of A and test whether $\sigma_{2^{K-1}+1}/\sigma_1 < \tau_2$
  end
  Decode $\widehat{\mathbf{q}}_j$ from the pattern of passing tests
end

return $\widehat{\mathbf{Q}}$

Step 2), followed by an eigendecomposition of a matrix of size $2^{|S|} \times 2^{|S|}$, which is constant in J. The cost of each test is therefore linear in J, with intermediate memory bounded by $O(4^K)$, and the full tensor $\mathcal{T}$ is never instantiated at any point after the TT cores are available.

4.4. Experiments

We evaluate the TT-based pipeline on simulated BN2A data and compare it against the original Gu (2026) algorithm applied to the explicit empirical tensor.

Setup. We fix $J = 15$ and $K = 3$, with a Q-matrix $\mathbf{Q}$ containing 6 rows with a single 1 (two for each hidden variable) and 9 rows with multiple 1s, ensuring the configuration is identifiable. Leak parameters $q_{j,0}$ are sampled uniformly from $\{0.70, 0.75, \ldots, 0.95\}$ and penalty parameters $q_{j,k}$ from $\{0.10, 0.20, 0.30, 0.40\}$. For each repetition we sample $n = 100{,}000$ response vectors under a uniform prior over the $2^K = 8$ hidden patterns, build the empirical tensor $\widehat{\mathcal{T}}$ with Laplace smoothing (additive constant 0.5), and apply both pipelines. The rank thresholds in Algorithm 1 are set to $\tau_1 = 0.15$ and $\tau_2 = 0.03$, following Gu (2026).

Results. Table 3 reports the Q-matrix recovery (structure learning) rate of both pipelines, measured at the entry level (fraction of correctly recovered entries of $\mathbf{Q}$ out of $J \cdot K = 45$)

Table 3: Q-matrix recovery on the empirical tensor $\widehat{\mathcal{T}}$ for $J = 15$, $K = 3$, $n = 100{,}000$. Both pipelines achieve perfect recovery; the TT-based pipeline performs all rank tests without materializing $\widehat{\mathcal{T}}$.

Pipeline	Entry-level	Row-level	Tests require full tensor?
Full tensor + Gu (baseline)	45/45	15/15	yes
TT cores + Gu (this work)	**45/45**	**15/15**	**no**

Table 4: Q-matrix recovery of the TT cores + Gu pipeline as a function of sample size n, with $J = 15$, $K = 3$. Entry-level and row-level columns report mean / best / worst across three independent replicates. The pipeline achieves perfect recovery for $n \geq 10{,}000$, well below $|\mathcal{T}| = 2^J = 32{,}768$.

n	$n/2^J$	Entry-level	Row-level	Step 1 failures
5,000	15%	43.3/45/40	13.3/15/10	0/3
10,000	31%	45.0/45/45	15.0/15/15	0/3
25,000	76%	45.0/45/45	15.0/15/15	0/3
50,000	153%	45.0/45/45	15.0/15/15	0/3
100,000	305%	45.0/45/45	15.0/15/15	0/3

and at the row level (fraction of fully correct rows out of $J = 15$). Both pipelines achieve perfect recovery, confirming that the TT-based rank tests are functionally equivalent to the original tests on the full tensor for this sample size.

Sample-size dependence. The result above uses $n = 100{,}000$, which exceeds $|\mathcal{T}| = 32{,}768$. To characterize how recovery depends on the sample size, we repeated the experiment for $n \in \{5{,}000, 10{,}000, 25{,}000, 50{,}000, 100{,}000\}$ with three independent replicates per value, keeping all other settings fixed (same Q-matrix $\mathbf{Q}$, same parameter distributions, canonical thresholds $\tau_1 = 0.15$ and $\tau_2 = 0.03$ with no per-run tuning). Table 4 reports the entry-level and row-level recovery aggregated across replicates.

Two observations are worth noting. First, perfect recovery in all three replicates is achieved as soon as $n \geq 10{,}000$, which corresponds to roughly one third of the tensor size. Second, even at $n = 5{,}000$ (only 15% of $|\mathcal{T}|$), the pipeline never fails at Step 1: pure-pair detection remains intact, and the only errors come from misclassification of two multi-attribute items in a single replicate. The pipeline therefore does not require sample sizes proportional to 2^J, despite operating on a smoothed empirical tensor.

Implementation and reproducibility. Experiments were performed in MATLAB on a MacBook Pro with a quad-core Intel Core i7 processor at 2.2 GHz. The TT-based pipeline completes the full structure learning (Q-matrix recovery) in under one second for this configuration. The baseline pipeline, which must explicitly build $\widehat{\mathcal{T}}$ (32,768 entries for

$J = 15$), is comparable at this scale; the asymptotic advantage of the TT-based pipeline appears for larger J, where the baseline is no longer tractable. Code is available at https://github.com/pragmaths/TTDforBN2A.

5. Concluding Remarks

This paper has connected two lines of work: the Tensor Train decomposition from numerical multilinear algebra, and the constructive Q-matrix identification algorithm of Gu (2026) from cognitive diagnostic modeling, equivalent to learning the structure of BN2A models. We have shown how the rank tests can be evaluated directly on TT cores, without materializing the joint probability tensor, and validated the pipeline experimentally at $J = 15$, $K = 3$, learning the structure exactly. Notably, recovery does not require sample sizes proportional to the tensor size: it is perfect at $n = 10{,}000$ (roughly 30% of $|\mathcal{T}|$) and degrades only mildly at $n = 5{,}000$.

We also applied the greedy mode-ordering algorithm of Tichavský and Straka (2025) to BN2A tensors. The BN2A formulation imposes no ordering constraint on the observed variables: permuting the modes of $\mathcal{T}$ merely permutes the rows of $\mathbf{Q}$, and since singular values of unfoldings are invariant under mode permutations, Gu's rank tests apply to any mode ordering, provided the item-position correspondence is tracked. On the population tensor at $J = 15$, $K = 3$, reordering reduces the TT parameter count from 1,104 to 960 (13%), with maximum bond dimension unchanged. An earlier version of our experiments reported degraded recovery under reordering; the cause was an order-sensitive ambiguity in Step 1 (pairs of multi-attribute items sharing a Q-row are numerically close to rank 2 and can displace true pure clusters). Ranking candidate clusters by their worst internal singular-value ratio removes the order sensitivity and restores exact recovery (45/45) under the compression-optimal ordering and under random reorderings alike. On the empirical tensor, however, sampling noise saturates the bond dimensions at the cap 2^K, so reordering brings no net storage gain.

While extending the experiments to $J > 16$ we observed that Step 2 degrades as J grows with K fixed: Step 1 continues to cluster added pure items correctly, but all multi-attribute items collapse to the same default Q-row. The cause is the marginalization implicit in Proposition 2 of Gu (2026), which sums over $J - 2K$ items and weakens the spectral signal as J grows; this is a property of the identification algorithm, not of the TT representation, and it limits the range of J for which the pipeline is effective at fixed K.

Three directions follow naturally: direct estimation of the TT cores from individual responses, bypassing the materialization of $\widehat{\mathcal{T}}$ entirely; modifications of the rank tests that mitigate the Step 2 marginalization issue; and extension to larger K, opening the pipeline to richer cognitive diagnostic models and to other bipartite latent-variable models with binary observations; when 2^K itself grows too large, truncating the TT-SVD at a cap below 2^K trades accuracy for tractability, and the effect of such truncation on the rank tests remains to be characterized.

Acknowledgments

The authors acknowledge financial support for this work: the first author was supported by the Czech Science Foundation through the project No. 25-18070S, and the third author was supported by the project "Research of Excellence on Digital Technologies and Wellbeing CZ.02.01.01/00/22_008/0004583" which is co-financed by the European Union.

References

Elizabeth S. Allman, Catherine Matias, and John A. Rhodes. Identifiability of parameters in latent structure models with many observed variables. *The Annals of Statistics*, 37 (6A):3099–3132, 2009.

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

Jing Chen, Song Cheng, Haidong Xie, Lei Wang, and Tao Xiang. Equivalence of restricted Boltzmann machines and tensor network states. *Physical Review B*, 97(8):085104, 2018. URL https://link.aps.org/doi/10.1103/PhysRevB.97.085104.

F. J. Díez and M. J. Druzdzel. Canonical Probabilistic Models for Knowledge Engineering. Technical Report CISIAD-06-01, UNED, Madrid, Spain, 2006. URL https://www.cisiad.uned.es/techreports/canonical.pdf.

Gaspard Ducamp, Philippe Bonnard, Anthony Nouy, and Pierre-Henri Wuillemin. An Efficient Low-Rank Tensors Representation for Algorithms in Complex Probabilistic Graphical Models. In Manfred Jaeger and Thomas Dyhre Nielsen, editors, *Proceedings of the 10th International Conference on Probabilistic Graphical Models*, volume 138 of *Proceedings of Machine Learning Research*, pages 173–184. PMLR, 23–25 Sep 2020. URL https://proceedings.mlr.press/v138/ducamp20b.html.

Yuqi Gu. Constructive Q-matrix Identifiability via Novel Tensor Unfolding. *Psychometrika*, 91(1):131–150, 2026. doi: 10.1017/psy.2025.10078.

Sarah Hartz and Louis Roussos. The Fusion Model for Skills Diagnosis: Blending Theory with Practicality. *ETS Research Report Series*, 2008(2):i–57, 2008. URL https://doi.org/10.1002/j.2333-8504.2008.tb02157.x.

Ivan V. Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33 (5):2295–2317, 2011. URL https://doi.org/10.1137/090752286.

Iván Pérez and Jiří Vomlel. On Identifiability of BN2A Networks. In Zied Bouraoui and Srdjan Vesic, editors, *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 136–148, Cham, 2024. Springer Nature Switzerland. ISBN 978-3-031-45608-4. URL https://link.springer.com/chapter/10.1007/978-3-031-45608-4_11.

Petr Savicky and Jiří Vomlel. Exploiting tensor rank-one decomposition in probabilistic inference. *Kybernetika*, 43(5):747–764, 2007. URL https://www.kybernetika.cz/content/2007/5/747.

Michael A. Shwe, Blackford Middleton, David E. Heckerman, Max Henrion, Eric J. Horvitz, Harold P. Lehmann, and Gregory F. Cooper. Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base: I. The probabilistic model and inference algorithms. *Methods of Information in Medicine*, 30(4):241–255, 1991.

Petr Tichavský and Ondřej Straka. Optimizing the Order of Modes in Tensor Train Decomposition. *IEEE Signal Processing Letters*, 32:1361–1365, 2025. doi: 10.1109/LSP.2025.3552005.

Jiří Vomlel and Petr Tichavský. Probabilistic inference with noisy-threshold models based on a CP tensor decomposition. *International Journal of Approximate Reasoning*, 55(4): 1072–1092, 2014. doi: https://doi.org/10.1016/j.ijar.2013.12.002.