Learning Polynomial Transformations via Generalized Tensor Decompositions*

Sitan Chen
Harvard SEAS & Berkeley EECS
Cambridge, MA, USA
sitan@seas.harvard.edu

Jerry Li
Microsoft Research
Redmond, WA, USA
jerrl@microsoft.com

Yuanzhi Li
Microsoft Research & CMU
Redmond, WA, USA
yuanzhil@andrew.cmu.edu

Anru Zhang
Duke Biostatistics & Bioinformatics
Durham, NC, USA
anru.zhang@duke.edu

ABSTRACT

We consider the problem of learning high dimensional polynomial transformations of Gaussians. Given samples of the form $f(x)$, where $x \sim \mathcal{N}(0, \mathrm{Id}_r)$ is hidden and $f : \mathbb{R}^r \to \mathbb{R}^d$ is a function where every output coordinate is a low-degree polynomial, the goal is to learn the distribution over $f(x)$. One can think of this as a simple model for learning deep generative models, namely pushforwards of Gaussians under two-layer neural networks with polynomial activations, though the learning problem is mathematically natural in its own right.

Our first main result is a polynomial-time algorithm for learning quadratic transformations of Gaussians in a smoothed setting. Our second main result is a polynomial-time algorithm for learning constant-degree polynomial transformations of Gaussian in a smoothed setting, when the rank of the associated tensors is small. In fact our results extend to any rotation-invariant input distribution, not just Gaussian. These are the first end-to-end guarantees for learning a pushforward under a neural network with more than one layer.

While our work aims to take an initial step towards understanding why generative models perform so well in practice, the algorithmic problems that we solve along the way are also of independent interest. We give the first provably efficient algorithms for *tensor ring decomposition*, a popular non-commutative generalization of tensor decomposition that is used in practice to implicitly store large tensors [107], as well as for a new variant of matrix factorization where the factors arise from low-rank tensors.

*Part of this work was done while SC, JL, and AZ were visiting the Simons Institute for the Theory of Computing. SC was supported in part by NSF Award 2103300. AZ was supported in part by NSF Award CAREER 2203741.

STOC '23, June 20–23, 2023, Orlando, FL, USA

ACM ISBN 978-1-4503-9913-5/23/06…$15.00
https://doi.org/10.1145/3564246.3585209

CCS CONCEPTS

• **Computing methodologies** → **Unsupervised learning**; **Neural networks**; • **Theory of computation** → **Semidefinite programming**.

KEYWORDS

Generative models, pushforwards, unsupervised learning, sum-of-squares, tensor ring decomposition, matrix product states

ACM Reference Format:
Sitan Chen, Jerry Li, Yuanzhi Li, and Anru Zhang. 2023. Learning Polynomial Transformations via Generalized Tensor Decompositions. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing (STOC '23), June 20–23, 2023, Orlando, FL, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3564246.3585209

1 INTRODUCTION

We consider the problem of learning a polynomial transformation. Suppose there is an unknown low-degree polynomial $f : \mathbb{R}^r \to \mathbb{R}^d$, where $r \ll d$, and we are given samples of the form $f(x_1), \ldots, f(x_n)$, where $x_1, \ldots, x_n$ are independent samples, not revealed to the learner, from some simple *seed distribution* D. The goal is to approximately learn the distribution of $f(x)$ for $x \sim D$, perhaps even by approximately recovering f. When f is linear and D is a product distribution, this is the well-studied problem of *independent component analysis* [49].

While this problem is natural in its own right, it also has relevance to the theory of deep generative models such as variational auto-encoders (VAEs) [76] and generative adversarial networks (GANs) [55]. Indeed, a polynomial transformation is nothing more than a generative model computed by a two-layer neural network with polynomial activations. Despite their apparent simplicity, pushforwards of simple distributions like Gaussians under such networks can already be used to generate images of nontrivial quality [83, Figure 2]. Understanding the learnability of such polynomial transformations can thus be thought of as a first step towards understanding the learnability of real-world generative models.

Unfortunately, despite the fundamental nature of this question, very little is known. The only provable results for this problem [83] only hold for extremely structured instances, and only under a certain conjectured structural result. To the best of our knowledge, to date, there are no algorithms with end-to-end provable guarantees for learning polynomial transformations of degree strictly

larger than 1. Indeed, more generally, there are no algorithms with such guarantees for learning pushforwards of neural networks with more than one layer in any non-trivial setting. Therefore, in this paper, we ask:

When can we provably learn polynomial transformations?

We will focus on the setting where $D = \mathcal{N}(0, \mathrm{Id})$, a standard choice of seed distribution in both the theoretical and applied generative modeling literature.

Tensor problems. Our approach to learning polynomial transformations is based on the method of moments. For example, in the special case where the output coordinates of f are computed by quadratic polynomials, i.e. $f(x) = (x^\top Q_1^* x, \ldots, x^\top Q_d^* x)$ for unknown symmetric matrices $Q_1^*, \ldots, Q_d^* \in \mathbb{R}^{r\times r}$, the task of recovering the parameters of f from the low-order moments of the polynomial transformation turns out to give rise to the following intriguing tensor problem (see Section 5):

There are unknown matrices $Q_1^*, \ldots, Q_d^* \in \mathbb{R}^{r\times r}$, and the goal is to recover them up to trivial symmetries, given noisy estimates for $\mathrm{Tr}(Q_a^* Q_b^*)$ and $\mathrm{Tr}(Q_a^* Q_b^* Q_c^*)$ for all a, b, c. When $\{Q_a^*\}$ are diagonal, this is equivalent to (degree-3) symmetric tensor decomposition (see Appendix in full version), a problem which has received considerable attention, especially in recent years, in the theoretical computer science and learning theory literature [16, 18, 51, 60, 64, 65, 82, 86]. The problem for general $\{Q_a^*\}$ is thus the natural "non-commutative" generalization of tensor decomposition.

In other communities, this tensor problem goes by the name of *tensor ring decomposition* [107]. Tensor ring decomposition, and related concepts such as hierarchical Tucker rank [15, 89] and tensor train decomposition [90, 91], were first proposed in the condensed matter physics community [101], and were later adopted in the neural network community as ways to concisely represent large tensors in a way which still allows for efficient linear algebraic computations [107]. Various heuristics have been proposed for this problem [75, 107], though to date, none of these come with provable guarantees for tensor ring decomposition in any nontrivial regime of parameters, and even in the noiseless setting where one has exact access to $\mathrm{Tr}(Q_a^* Q_b^*)$ and $\mathrm{Tr}(Q_a^* Q_b^* Q_c^*)$. This is in stark contrast to the state of affairs with traditional tensor decomposition, where for many settings, often in the presence of considerable noise, there are many polynomial time algorithms with provable guarantees. This begs the natural question:

When can we efficiently solve tensor ring decomposition?

While this is of tremendous interest in its own right, our interest comes from the fact that this is fundamentally related to learning quadratic transformations of Gaussians. Indeed, recovering the parameters of such a distribution from its moments of degree at most 3 is exactly equivalent to solving tensor ring decomposition (see Section 5). Understanding tensor ring decomposition thus seems like an important step towards understanding our central learning problem.

We also give algorithms based on method of moments for learning *higher-degree* polynomial transformations. Naturally, this gives rise to other tensor problems of even greater complexity, and we defer a discussion of these to Sections 2.1 and 5. We view the design of algorithms for tensor ring decomposition and these other tensor problems as one of our primary contributions.

1.1 Our Contributions

In this paper, we give the first efficient algorithms for learning high dimensional polynomial transformations of Gaussians, under mild non-degeneracy conditions that we demonstrate are satisfied with negligible failure probability in reasonable smoothed analysis settings. Along the way, we also provide the first efficient algorithms for tensor ring decomposition and related tensor problems under analogous conditions.

Efficient algorithms for quadratic transformations and tensor ring decomposition. Our first result is a polynomial time algorithm for learning smoothed (homogeneous) quadratic transformations of Gaussians, in sufficiently high dimensions:

Theorem 1.1 (Informal). *For any $d \in \mathbb{N}$ sufficiently large and any $\varepsilon > 0$, $1/\mathrm{poly}(d)$-smoothed quadratic transformations of Gaussian with input dimension $r = \widetilde{O}(\sqrt{d})$ are learnable (both in parameter distance and Wasserstein distance) to error ε in $\mathrm{poly}(r, 1/\varepsilon) \cdot d$ time and $\mathrm{poly}(r, 1/\varepsilon)$ samples with probability at least $1 - \exp(-\mathrm{poly}(r))$ over the smoothing.*

To the best of our knowledge, this is the first end-to-end provable algorithmic result for learning pushforwards given by a neural network with more than a single layer (see Section 4 for further discussion). Note that the condition $r = \widetilde{O}(\sqrt{d})$ here means that the pushforward distribution is supported on a low-dimensional manifold, which is quite natural in practice [92].

Our smoothed model is the standard one in which the instance is given by a small random perturbation of a worst-case instance (see Section 2). As with many results in smoothed analysis, our results hold more generally under mild deterministic non-degeneracy conditions.

We remark that while our guarantee holds for all d which are *at least* quadratic in r, standard dimension reduction arguments (see full version) show that the hardest case is really when d is exactly quadratic in r. This is the threshold at which the d polynomials defining the transformation first span the space of quadratic polynomials in r variables.

We complement Theorem 1.1 with an information-theoretic lower bound (see Appendix in the full version), which states that in the worst case, parameter learning for quadratic transformations requires exponentially many samples, even in one dimension. Combined with computational hardness results for improper density estimation of worst-case ReLU network transformations of Gaussians [28–30], this suggests that some beyond-worst-case assumptions are necessary to obtain efficient algorithms. Intuitively, our non-degeneracy assumptions give us a "blessing of dimensionality" phenomenon which allows us to obtain multiple linearly independent "views" of the underlying transformation.

Theorem 1.1 is based on the following new algorithm for tensor ring decomposition:

Theorem 1.2 (Informal, see Theorem 6.3). *For any $d \in \mathbb{N}$ sufficiently large and any $\varepsilon > 0$, given a $1/\mathrm{poly}(d)$-smoothed instance of ε-noisy tensor ring decomposition in dimension $r = \widetilde{O}(\sqrt{d})$, there is a polynomial time algorithm which recovers the unknown matrices to error $\mathrm{poly}(\varepsilon, r)$ up to trivial symmetries in $\mathrm{poly}(r, 1/\varepsilon) \cdot d$ time with probability at least $1 - \exp(-\mathrm{poly}(r))$ over the smoothing.*

Our algorithms for Theorems 1.1 and Theorems 1.2 are based on the Sum-of-Squares (SoS) "proofs to algorithms" framework, which in recent years has been applied to solve a number of high-dimensional statistical problems. However, the design of our algorithm differs quite substantially from prior techniques used within this literature. As we explain in Section 3, the $\mathrm{Tr}(Q_a^* Q_b^*)$'s in tensor ring decomposition give us the unknown $r \times r$ matrices $Q_1^*, \dots, Q_d^*$, up to a shared, unknown rotation, but as vectors in r^2 dimensions. The heart of our algorithm is an SoS proof that the only such rotations which can additionally match the $\mathrm{Tr}(Q_a^* Q_b^* Q_c^*)$'s are in fact Kronecker powers of $r \times r$-dimensional rotation matrices. In other words, up to gauge symmetry in the $r \times r$-dimensional space, the $r^2 \times r^2$-dimensional rotations which respect our constraints are unique, and moreover, SoS witnesses this fact. Consequently, this implies that we can search for these rotations using an SoS program, and the result can be easily rounded to solve the overall problem.

Efficient algorithms for low-rank polynomial transformations. For our final result, we turn to polynomial transformations of higher degree. We show that (homogeneous) polynomial transformations of odd constant degree can be learned efficiently, as long as the transformation can be represented using low rank tensors. Recall that any homogeneous degree ω polynomial $p : \mathbb{R}^r \to \mathbb{R}$ can be associated with a symmetric tensor $T : \mathbb{R}^r \to (\mathbb{R}^r)^{\otimes \omega}$, so that $p(x) = \langle T, x^{\otimes \omega} \rangle$. We say that a polynomial is rank ℓ if the associated tensor has symmetric rank ℓ, and we say that a polynomial transformation $f : \mathbb{R}^r \to \mathbb{R}^d$ has rank ℓ, if each output coordinate is computed by a rank-ℓ polynomial. From the perspective of neural networks, ℓ corresponds to the *channels* of the hidden layer per neuron. Our main result here is:

Theorem 1.3 (Informal). *There is an absolute constant $c > 0$ such that for any $d \in \mathbb{N}$ sufficiently large and any $\varepsilon > 0$, $1/\mathrm{poly}(d)$-smoothed rank-$\ell = O(1)$ transformations of odd degree $\omega = O(1)$ with seed length $r = \widetilde{O}(d^{c/(\omega \ell)})$ are learnable (both in parameter distance and Wasserstein distance) to error ε in $\mathrm{poly}(r, 1/\varepsilon) \cdot d$ time and $\mathrm{poly}(r, 1/\varepsilon)$ samples with probability at least $1 - \exp(-\mathrm{poly}(r))$ over the smoothing.*

At its heart, our algorithm follows the same rough structure as the one for the quadratic case, that is, we must show in SoS that the unknown rotation over r^ω dimensions which maps the ground truth to our estimates must arise as a Kronecker power of a rotation over r dimensions. However, the arguments here are much more subtle. For starters, for a high-degree polynomial transformation, even the low-order moments are unwieldy even to write down, let alone work with.

For this reason, unlike in the quadratic case, here we only work with second-order moments. In place of tensor ring decomposition, this leads to a new inverse problem that we call *low-rank factorization*, which may be of independent interest: given unknown low-rank symmetric tensors $T_1^*, \dots, T_d^*$, recover them from estimates of every $\langle T_a^*, T_b^* \rangle$ up to the trivial $r \times r$ rotational symmetry (see Definition 2.8). A priori it is unclear why this should be possible, e.g. if $T_1^*, \dots, T_d^*$ weren't constrained to be low-rank, then one could only hope to recover them up to a global $r^\omega \times r^\omega$ rotation. We show that surprisingly, the low-rank constraints force this rotation to be the Kronecker power of an $r \times r$ rotation. The proof of this is the most involved part of this work: the difficulty comes in large part from the fact that symmetric tensor rank, unlike matrix rank, is notoriously difficult to capture using simple polynomial constraints [80]. We refer the reader to Section 3 for more details.

Finally, we remark that all of our guarantees for learning transformations (Theorem 1.1 and 1.3) in fact hold for transformations of *any* rotation-invariant seed distribution with suitable moment bounds (see Sections 5.1 and 5.2), not just of $\mathcal{N}(0, \mathrm{Id}_r)$.

2 GENERATIVE MODEL AND INVERSE PROBLEMS

In this section, we formally define the models we study throughout this paper.

Definition 2.1 (Polynomial Transformations). *For $\omega \geq 2$, a d-dimensional degree-ω transformation with seed length r is a distribution $\mathcal{D}$ over $\mathbb{R}^d$ specified by tensors $T_1^*, \dots, T_d^* \in (\mathbb{R}^r)^{\otimes d}$. To sample from $\mathcal{D}$, one samples $x \sim \mathcal{N}(0, \mathrm{Id}_r)$ and outputs*

$$(\langle T_1^*, x^{\otimes \ell} \rangle, \dots, \langle T_d^*, x^{\otimes \ell} \rangle).$$

Equivalently, $\mathcal{D}$ is the pushforward *of the standard Gaussian measure on $\mathbb{R}^r$ under the map $x \mapsto (\langle T_1^*, x^{\otimes \ell} \rangle, \dots, \langle T_d^*, x^{\otimes \ell} \rangle)$.*

We will collectively refer to the tensors $T_1^*, \dots, T_d^*$ as the polynomial network *specifying $\mathcal{D}$. If $\omega = 2$, we will use $Q_1^*, \dots, Q_d^* \in \mathbb{R}^{r \times r}$ in place of $T_1^*, \dots, T_d^*$. If $T_1^*, \dots, T_d^*$ are of rank ℓ, then we say that $(T_1^*, \dots, T_d^*)$ is a* rank-ℓ polynomial network.

We will study the learnability of polynomial transformations in the following smoothed analysis settings. For quadratic transformations, we consider entrywise Gaussian perturbations.

Definition 2.2 (Smoothed Quadratic Networks). *Let $\rho > 0$. We say a degree-2 polynomial network $Q_1^*, \dots, Q_d^* \in \mathbb{R}^{r \times r}$ is ρ-*fully-smoothed *if $Q_1^*, \dots, Q_d^*$ were generated as follows: for some (possibly worst-case) matrices $\overline{Q}_1, \dots, \overline{Q}_d \in \mathbb{R}^{r \times r}$, each Q_a^* is obtained by independently sampling a symmetric matrix G_a whose diagonal and upper triangular entries are independent draws from $\mathcal{N}(0, 1)$ and forming $Q_a^* \triangleq \overline{Q}_a + \frac{\rho}{r} \cdot G_a$. We refer to the matrices $\overline{Q}_1, \dots, \overline{Q}_d$ as the* base network.

For low-rank transformations, we consider perturbations of the rank-1 *tensor components*.

Definition 2.3 (Smoothed Low-Rank Networks). *Let $\rho > 0$. We say that a rank-ℓ polynomial network $T_1^*, \dots, T_d^* \in (\mathbb{R}^r)^{\otimes \omega}$ is ρ-*componentwise-smoothed *if $T_1^*, \dots, T_d^*$ were generated as follows: for some (possibly worst-case) r-dimensional vectors $\{\overline{v}_{a,i}\}_{a \in [d], i \in [\ell]}$, each T_a^* is obtained by independently sampling $g_{a,1}, \dots, g_{a,\ell}$ from $\mathcal{N}(0, \mathrm{Id}_r)$ and forming $T_a^* \triangleq \sum_{i=1}^{\ell} (\overline{v}_{a,i} + \frac{\rho}{\sqrt{r}} \cdot g_{a,i})^{\otimes \omega}$. Similar to Definition 2.2, we refer to the tensors $\overline{T}_1, \dots, \overline{T}_d$ given by $\overline{T}_a \triangleq \sum_{i=1}^{\ell} v_{a,i}^{\otimes \ell}$ as the* base network.

In this paper we give guarantees for parameter learning polynomial transformations. There are some basic symmetries to be aware of. First, if T and T' differ by a skew-symmetric form, that is if $\sum_{\pi \in S_\omega} (T - T')_{i_{\pi(1)} \cdots i_{\pi(\omega)}} = 0$ for all $i_1, \dots, i_\omega \in [r]$, then $\langle T, x^{\otimes \omega} \rangle$ and $\langle T', x^{\otimes \omega} \rangle$ are identical as polynomials in x. For this reason, we will henceforth assume without loss of generality that the network $T_1^*, \dots, T_d^*$ consists of *symmetric tensors*.

Additionally, because the seed distribution $\mathcal{N}(0, \mathrm{Id}_r)$ that is being pushed forward through the polynomial network is rotation-invariant, the network of a polynomial transformation is only identifiable up to a *gauge symmetry*. Let $O(r)$ denote the group of orthogonal $r \times r$ matrices. Given a tensor $T \in (\mathbb{R}^r)^{\otimes\omega}$ and orthogonal matrix $U \in O(r)$, define the tensor $F_U(T) \in (\mathbb{R}^r)^{\otimes\omega}$ by

$$F_U(T)_{i_1 \cdots i_\omega} = \sum_{j_1,\ldots,j_\omega \in [r]} U_{i_1 j_1} \cdots U_{i_\omega j_\omega} T_{j_1 \cdots j_\omega} \tag{1}$$

for all $i_1, \ldots, i_\omega \in [r]$. Note that when $\omega = 2$ so that T is an $r \times r$ matrix, then $F_U(T) = UTU^\top$. The following is immediate:

Lemma 2.4 (Gauge symmetry)**.** *For any network $T_1^*, \ldots, T_d^* \in (\mathbb{R}^{\otimes r})^{\otimes\omega}$ and any orthogonal matrix $U \in O(r)$, the transformation specified by the polynomial network $T_1^{**}, \ldots, T_d^{**}$, where $T_a^{**} \triangleq F_{U^{\otimes\omega}}(T_a^*)$ is identical in distribution to the one specified by $T_1^*, \ldots, T_d^*$.*

In this work we study parameter learning and thus formulate this learning task as recovering the polynomial network modulo this freedom:

Definition 2.5 (Parameter Distance)**.** *Given polynomial networks $T_1, \ldots, T_d$ and $T_1', \ldots, T_d'$, define* parameter distance $d_{\mathrm{G}}(\{T_a\}, \{T_a'\})$ *by $d_{\mathrm{G}}(\{T_a\}, \{T_a'\}) \triangleq \min_{U \in O(r)} \max_{a \in [d]} \|F_{U^{\otimes\omega}}(T_a) - T_a'\|_F$.*

It is not hard to see that parameter learning implies proper density estimation.

We note that in general it is not true that the parameters of a polynomial transformation must be uniquely determined up to gauge symmetry. For example, it was shown in [57] that there exist cubic polynomials $p, q : \mathbb{R}^2 \to \mathbb{R}$ for which the corresponding pushforwards of $\mathcal{N}(0, \mathrm{Id}_2)$ are identical as distributions, but for which p and q are *not* equivalent up to gauge symmetry. Nevertheless, the fact that we are able to show parameter learning up to gauge symmetry is possible in smoothed settings suggests that such examples are quite pathological.

2.1 Inverse Problems

Our algorithms for learning polynomial transformations are based on the method of moments. In general, the intricate combinatorial structure of the higher-order moments of a polynomial transformation makes them quite difficult to work with, especially when the degree of the transformation itself is large. In this work however, we show that for smoothed networks, it suffices to work with moments up to degree at most three. That is, we show how to recover the parameters of a smoothed polynomial transformation $\mathcal{D}$ using only estimates of the form $\mathbb{E}[z_a z_b z_c], \mathbb{E}[z_a z_b], \mathbb{E}[z_a]$ for $z \sim \mathcal{D}$. As we show in Section 5, these moments take a particular form so that the problem of reconstructing parameters from moments naturally gives rise to the following inverse problems.

Definition 2.6 (Tensor Ring Decomposition)**.** *Let $\eta > 0$, and let $Q_1^*, \ldots, Q_d^* \in \mathbb{R}^{r \times r}$ be unknown symmetric matrices. Given as input a matrix $S \in \mathbb{R}^{d \times d}$ and a tensor $T \in \mathbb{R}^{d \times d \times d}$ satisfying*

$$\left|\mathrm{Tr}(Q_a^* Q_b^*) - S_{a,b}\right| \le \eta$$
$$\left|\mathrm{Tr}(Q_a^* Q_b^* Q_c^*) - T_{a,b,c}\right| \le \eta$$

$a, b, c \in [d]$, the goal is to output matrices $\widehat{Q}_1, \ldots, \widehat{Q}_d$ for which the parameter distance $d_{\mathrm{G}}(\{Q_a^*\}, \{\widehat{Q}_a\})$ is small.

Remark 2.7. *This is slightly different from how tensor ring decomposition is traditionally posed [107] as usually one only assumes that T is given. For learning polynomial transformations however, it is easy to get access to both S and T, so we work with Definition 2.6.*

This specializes to the well-studied problem of symmetric tensor decomposition when Q_a^* are diagonal: if $v_i \in \mathbb{R}^d$ denotes the vector with a-th entry $(Q_a^*)_{ii}$, then $T \approx \sum_i v_i^{\otimes 3}$ (see full version for details).

We also study the following (to our knowledge, new) variant of matrix factorization:

Definition 2.8 (Low-Rank Factorization)**.** *Let $\eta > 0$, and let T_1^*, ..., $T_d^* \in (\mathbb{R}^r)^{\otimes\omega}$ be unknown symmetric tensors of rank ℓ. Given a known positive definite matrix $\Sigma \in \mathbb{R}^{r^\omega \times r^\omega}$, let $\langle \cdot, \cdot \rangle_\Sigma$ denote the associated inner product. Given as input a matrix $S \in \mathbb{R}^{d \times d}$ satisfying*

$$\left|\langle \mathrm{vec}(T_a^*), \mathrm{vec}(T_b^*) \rangle_\Sigma - S_{a,b}\right| \le \eta \ \ \forall\, a, b \in [d],$$

the goal is to output $\widehat{T}_1, \ldots, \widehat{T}_d$ for which $d_{\mathrm{G}}(\{T_a^*\}, \{\widehat{T}_a\})$ is small.

A priori, it is not even clear that such a recovery guarantee is possible. Indeed, without the extra condition that $T_1^*, \ldots, T_d^*$ are low rank, the recovery goal in Definition 2.8 is impossible, even for $\eta = 0$ and $\Sigma = \mathrm{Id}$. In that case, the constraints $\langle T_a^*, T_b^* \rangle = S_{a,b}$ at best specify $\{T_a^*\}$ up to an $r^\omega \times r^\omega$ *rotation*, whereas in Definition 2.8 we are interested in recovery up to an $r \times r$ rotation!

In view of our application to polynomial transformations, we will be interested in Σ given by $\Sigma = \mathbb{E}_{x \sim D}[\mathrm{vec}(x)^{\otimes\omega} \mathrm{vec}(x)^{\otimes\omega\top}]$ for rotation-invariant distributions D over $\mathbb{R}^r$, e.g. $D = \mathcal{N}(0, \mathrm{Id})$.

3 TECHNICAL OVERVIEW

In this section we give a high-level overview of the key algorithmic ideas in this work. As our reduction from polynomial pushforwards to the inverse problems defined in Section 2.1 is straightforward (see Section 5), here we focus on describing our algorithms for the inverse problems, namely tensor ring decomposition and low-rank factorization. For both of these, we will sketch how to prove that the underlying parameters ($\{Q_a^*\}$ and $\{T_a^*\}$ respectively) are information-theoretically *identifiable* from the input, modulo gauge symmetry. As we show partially in Sections 6 and completely in the full version, with significant care, these proofs of identifiability can be implemented in the SoS proof system and thus yield efficient algorithms; we discuss the main challenges for doing so at the end of this overview. Along the way, we also discuss why existing approaches in the tensor decomposition literature like simultaneous diagonalization appear to fall short of solving the tensor problems we consider.

For simplicity, in this overview we focus on the noiseless setting, i.e. when $\eta = 0$ in Definitions 2.6 and 2.8, though in later sections we prove our guarantees for general η.

Overview notation. Subscripts/superscripts denote row/column indices for matrices. Given $Q \in \mathbb{R}^{r \times r}$, $\mathrm{vec}(Q) \in \mathbb{R}^{r^2}$ denotes its flattening. $e_i \in \mathbb{R}^r$ denotes the i-th standard basis vector.

3.1 Tensor Ring Decomposition

Hidden $r^2 \times r^2$ rotation. Recall that in tensor ring decomposition, there are unknown symmetric matrices $Q_1^*, \ldots, Q_d^*$, and we

want to recover them up to gauge symmetry given $\mathrm{Tr}(Q_a^* Q_b^*)$ and $\mathrm{Tr}(Q_a^* Q_b^* Q_c^*)$ for all a, b, c.

First, as discussed above, the only information the $\mathrm{Tr}(Q_a^* Q_b^*)$'s provide is the angle between every pair of matrices *regarded as an r^2-dimensional vector*. In particular, given only the $\mathrm{Tr}(Q_a^* Q_b^*)$'s, the best we can hope for is to estimate $\{Q_a^*\}$ up to an $r^2 \times r^2$ *rotation*[1] (see Section 6.3). More formally, we can only hope to produce matrices $Q_1, \ldots, Q_d$ for which there exists some $r^2 \times r^2$ orthogonal matrix U satisfying $U \,\mathrm{vec}(Q_a^*) = \mathrm{vec}(Q_a)$ for $a = 1, \ldots, d$. Recalling (1), we denote this by

$$F_U(Q_a^*) = Q_a \tag{2}$$

An example of such a U would be one corresponding to an $r \times r$ *rotation*. That is, consider a $r \times r$ orthogonal matrix V. We can check that the transformation sending any Q to $VQV^\top$ can be expressed in terms of F_U for U given by the *Kronecker square of V*. That is, if we index the rows and columns of U by $[r] \times [r]$ and let the (i, j)-th column be given by the flattening of $V^i (V^j)^\top$, then $F_U(Q) = VQV^\top$. In this case, we say that U *arises from* V.

Note that U's of this form comprise a vanishing fraction of all $r^2 \times r^2$ orthogonal matrices. The bulk of our analysis is thus centered around proving that the remaining *third-order* constraints in tensor ring decomposition, i.e. the $\mathrm{Tr}(Q_a^* Q_b^* Q_c^*)$'s, force U to take this special form.

Using third-order constraints. Note that we can interpret the $\mathrm{Tr}(Q_a^* Q_b^* Q_c^*)$'s as telling us the angle between the vectors $\mathrm{vec}(Q_a^*)$ and $\mathrm{vec}(Q_b^* Q_c^*)$ for any a, b, c. Using this, we can ensure that in addition to U sending every Q_a^* to Q_a, U also sends every $Q_b^* Q_c^*$ to $Q_b Q_c$, i.e.

$$F_U(Q_b^* Q_c^*) = Q_b Q_c.$$

To unpack what additional information this implies about U, let us pretend for a moment that $Q_1^*, \ldots, Q_d^*$ consisted of the matrices $\{E_{ij}\}$, where $E_{ij} = e_i e_j^\top$. For any i, j, we will refer to the corresponding Q_a as Q_{ij} so that $Q_{ij} = F_U(E_{ij})$. Note that $F_U(E_{ij})$ is the (i, j)-th column of U, reshaped into an $r \times r$ matrix. We will refer to this as U^{ij}.

Now what do the constraints 3.1 tell us? For any $i, j, j', k \in [r]$, note that $E_{ij} E_{j'k} = \mathbb{1}[j = j'] \cdot E_{ik}$. So the fact that $Q_{ij} Q_{j'k} = F_U(E_{ij} E_{j'k})$ implies that

$$U^{ij} U^{j'k} = \mathbb{1}[j = j'] \cdot U^{ik}. \tag{3}$$

It turns out that even if $\{Q_a^*\}$ are not given by $\{E_{ij}\}$, under mild conditions on $\{Q_a^*\}$ that hold in the smoothed setting (Part 2 of Assumption 1), U will still satisfy (3).

Using the relations (3). We now sketch how to argue, using the relations (3), that U must arise from an $r \times r$ rotation. Recall this means we must argue that the matrices U^{ij} are each given by the outer product of a pair of columns of some orthogonal matrix.

The main step is to argue that the matrices U^{ij} are rank-1 matrices. From (3), we have that $U^{ij} = U^{ii} U^{ij}$. Right-multiplying by ${U^{ij}}^\top$ on both sides and taking traces, we get

$$\begin{aligned} \mathrm{Tr}(U^{ij} {U^{ij}}^\top) &= \mathrm{Tr}(U^{ii} U^{ij} {U^{ij}}^\top) \\ &\le \|U^{ii}\|_F \|U^{ij} {U^{ij}}^\top\|_F = \|U^{ij} {U^{ij}}^\top\|_F, \end{aligned} \tag{4}$$

where in the third step we used the fact that U is orthogonal to conclude that $\|U^{ii}\|_F = 1$. As $U^{ij} {U^{ij}}^\top$ is psd, the above inequality holds with equality, so $U^{ij} {U^{ij}}^\top$ is a rank-1 matrix, implying that U^{ij} is as well.

Having showed there exist unit vectors $\{v_{ij}, w_{ij}\}$ for which $U^{ij} = v_{ij} w_{ij}^\top$, we can use (3) to conclude. For instance, (3) implies that $(U^{ii})^2 = U^{ii}$, so $v_{ii} = w_{ii}$. It also tells us that $U^{ii} U^{jj} = 0$ for $i \neq j$, so $\{v_{ii}\}$ are orthonormal. Lastly, it tells us that $U^{ii} U^{ij} = U^{ij}$ and $U^{ij} U^{jj} = U^{ji}$, so $v_{ij} = v_{ii}$ and $w_{ij} = v_{jj}$. These show that U arises from an $r \times r$ rotation with columns $\{v_{ii}\}$.

A catch: working with symmetric matrices. Thus far, an important detail that we have elided is that because $Q_1^*, \ldots, Q_d^*$ are symmetric, there is actually some ambiguity in how to define the $r^2 \times r^2$ matrix U mapping every Q_a^* to Q_a. For instance, given any such U, we could interchange the (i, j)-th and (j, i)-th columns (or more generally, replace them with arbitrary affine combinations of each other) and get a new matrix with the same property.

To resolve this ambiguity, we insist that U satisfy $U^{ij} = U^{ji}$ for every $i = j$. Unfortunately, this comes at a cost: U is no longer orthogonal. Additionally, because $\{Q_a^*\}$ are symmetric and thus span a smaller space than $\{E_{ij}\}$, we end up with weaker relations than (3).

We nevertheless show how to use these weaker relations to bootstrap a new matrix out of U with all the desired properties from the discussion above, i.e. orthogonality, (2), and (3).

Failure of other approaches. Given that tensor ring decomposition is a direct generalization of tensor decomposition, one might wonder whether there are straightforward ways to tailor off-the-shelf algorithms for the latter to our setting. While it is unclear how to adapt algorithms like tensor power method to tensor ring decomposition, one promising candidate is Jennrich's algorithm [60, 82]. Unfortunately, it turns out that because $\{Q_a^*\}$ are constrained to be generic *symmetric* matrices rather than generic matrices (a distinction which is irrelevant in the special case where $\{Q_a^*\}$ are diagonal which corresponds to tensor decomposition), the fact that $\{Q_a^*\}$ don't span all of $\mathbb{R}^{r \times r}$ breaks natural ways of adapting approaches based on simultaneous diagonalization. As discussed in the preceding paragraphs, the constraint that $\{Q_a^*\}$ must be symmetric, which is necessary for avoiding one of the trivial symmetries inherent in the problem, is a key technical hurdle that we overcome.

3.2 Low-Rank Factorization

We now turn to our second inverse problem. Here, while we also employ the general strategy of showing some unknown rotation between the ground truth $\{T_a^*\}$ to our estimates $\{T_a\}$ arises from an $r \times r$ rotation, there are a number of essential differences in how we implement this approach.

Hidden mapping respecting Σ norm. Recall there are unknown symmetric tensors $\{T_a^*\}$ of symmetric rank ℓ, and we want to recover them up to gauge symmetry given $\langle T_a^*, T_b^* \rangle_\Sigma$. While the Σ-norm is no longer Euclidean, we can still readily obtain estimates $\{T_a\}$ which agree with $\{T_a^*\}$ up to some $r^\omega \times r^\omega$ map U preserving the Σ-norm, and as before, we want to show that U arises from an

[1] Technically this is not quite true as $\{Q_a^*\}$ do not span the space of all $r \times r$ matrices as they are symmetric. We defer the discussion of how we circumvent this issue to later in the overview.

$r \times r$ rotation. If we index the rows and columns of U by $[r]^\omega$, this amounts to showing that there is some orthogonal $V \in \mathbb{R}^{r \times r}$ such that every column of U is given by some $\mathrm{vec}(V^{i_1} \otimes \cdots \otimes V^{i_\omega})$.

Rank-ℓ-preserving transformations. The key challenge that arises in low-rank factorization and not tensor ring decomposition is that we only have access to *pairwise* information about $\{T_a^*\}$. In the absence of third-order constraints which might have helped us to prove an identity like (3), we need to exploit the assumption that the unknown tensors $\{T_a^*\}$ are low-rank.

In particular, the fact that $\{T_a^*\}$ are low-rank and the fact that the estimates T_a that we output should also be low-rank places nontrivial constraints on U. Intuitively, because $\{T_a^*\}$ are "random-looking" in the smoothed setting, if d is sufficiently large then we expect that U should send *any* rank-ℓ tensor to a rank-ℓ tensor. Reasoning about this, especially in a way amenable to efficient algorithms, is quite delicate, because tensor rank is notoriously worse-behaved than matrix rank. We sidestep this by devising a relaxed notion of tensor rank that plays nicely with our algorithmic techniques (see Section 3.3) and show that U is "rank-ℓ-preserving." That is, it sends any tensor of symmetric rank ℓ to a tensor with "relaxed rank" ℓ.

Rank-ℓ-preserving implies rank-1-preserving. The reason it is useful for U to be rank-ℓ-preserving is that, as we show in the full version, it additionally implies that U is rank-$(\ell-1)$-preserving and thus, by induction, rank-1 preserving! Before we process the implications of the latter, we sketch the argument. For simplicity, suppose here that $r = \ell + 1$ and consider the special case of $\omega = 2$, where our relaxed notion of rank agrees with symmetric rank (i.e. matrix rank), though the argument also extends to any $\omega > 2$.

Starting with any rank-$(\ell-1)$ matrix $M \in \mathbb{R}^{(\ell+1)\times(\ell+1)}$, consider some rank-1 perturbation $c \cdot zz^\top$ that we will vary. By assumption, $F_U(M + c \cdot zz^\top) = F_U(M) + F_U(c \cdot zz^\top)$ has rank ℓ, so $\det(F_U(M) + F_U(c \cdot zz^\top)) = 0$. Formally differentiating this with respect to c at $c = 0$ yields

$$\sum_{i=1}^{\ell+1} \det\left(\; F_U(M)^{1:i-1} \;\middle|\; F_U(zz^\top)^i \;\middle|\; F_U(M)^{i+1:\ell+1} \; \right) = 0, \qquad (5)$$

where $A^{i:j}$ denotes the matrix consisting of the i-th to j-th columns of A. In particular, we can take the Laplace expansion of the i-th determinant in (5) along the i-th column, and (5) then becomes a linear combination of all $\ell \times \ell$ minors of $F_U(M)$, where the coefficients of this linear combination are given by entries of $F_U(zz^\top)$. By taking many choices of z, we can ensure that sufficiently many different linear combinations of these minors vanish to imply that the minors themselves vanish. This shows that $F_U(M)$ is rank-$(\ell-1)$ as desired.

Using rank-1-preservation to conclude. It turns out our relaxed notion of rank aligns with symmetric rank for rank-1 tensors, so rank-1-preservation implies U sends any symmetric rank-1 tensor to a symmetric rank-1 tensor. Note that if U mapped *any rank-1 tensor* (not necessarily symmetric) to a rank-1 tensor, then U would map any $e_{i_1} \otimes \cdots \otimes e_{i_\omega}$ to a rank-1 tensor, implying that the columns of U are flattenings of rank-1 tensors. At this point we would practically be done. Indeed, with some work, one could combine this with the fact that U preserves the Σ-norm to conclude that U indeed arises from an $r \times r$ rotation.

On the other hand, using only the fact that U sends *symmetric* rank-1 tensors to symmetric rank-1 tensors to show the same is far more involved and out of the scope of this overview. We give the full details in the full version.

3.3 Sum-of-Squares Algorithms

Proofs to algorithms. Our general approach for getting an algorithm out of all of this follows the usual SoS proofs-to-algorithms pipeline for statistical problems [62]. While our setting introduces a multitude of new conceptual twists to implementing this approach which we will explain below, we begin by outlining the basic setup. First, we introduce SoS variables $\{Q_a\}$ (resp. $\{T_a\}$) corresponding to our estimates for the ground truth $\{Q_a^*\}$ (resp. $\{T_a^*\}$) and constrain them to possess the same properties as the ground truth. For instance, for low-rank factorization, we require that $\{T_a\}$ are symmetric and satisfy $\langle T_a, T_b \rangle_\Sigma = \langle T_a^*, T_b^* \rangle_\Sigma$, and to constrain them to be low-rank, we also introduce SoS variables $\{v_{a,t}\}_{a \in [d], t \in [\ell]}$ and insist that $T_a = \sum_{t=1}^{\ell} v_{a,t}^{\otimes \ell}$. The hope is to turn the arguments above into a low-degree SoS proof that $\{T_a\}$ and $\{T_a^*\}$ are equivalent up to gauge symmetry, and then to apply some simple rounding procedure to a pseudoexpectation satisfying the aforementioned constraints to extract estimates for $\{T_a^*\}$.

This raises a number of challenges. How do we capture the $r^\omega \times r^\omega$ transformation U from the preceding discussion in SoS? How do we encode the condition that U has the structure of a Kronecker power of an $r \times r$ orthogonal matrix? And how do we actually round, given that everything is only specified up to gauge symmetry?

Implementing U as an SoS variable. For simplicity, we illustrate this in the setting of tensor ring decomposition. Having imposed the constraints $\mathrm{Tr}(Q_a Q_b) = \mathrm{Tr}(Q_a^* Q_b^*)$, we can rewrite these constraints as the matrix equality

$$NN^\top = N^* N^{*\top},$$

where $M, M^* \in \mathbb{R}^{d \times r^2}$ have a-th row given by $\mathrm{vec}(Q_a)$ and $\mathrm{vec}(Q_a^*)$ respectively. A linear transformation mapping every Q_a^* into Q_a can be thought of as a matrix U for which $UN^{*\top} = N^\top$. A natural way to construct such a matrix U would be to define $U = N^{-1} N^{*\top}$. Note that because $NN^\top = N^* N^{*\top}$, it would follow that U is an orthogonal matrix.

Of course this doesn't quite work as N is an SoS variable and thus does not have a left-inverse, but this is easy to remedy by introducing an additional variable L to the SoS program corresponding to this left-inverse and requiring that $LN = \mathrm{Id}$. We could then define U to be $LN^{*\top}$. We emphasize that U should not be thought of as another variable in our SoS program; after all, N^* is unknown to the algorithm designer, so the entries of U are merely unknown linear forms in the SoS variable L. For this reason, U is only referenced throughout the analysis of our SoS relaxation.

Finally, as discussed at the end of Section 3.1, there are some subtleties as N has repeated columns because $\{Q_a\}$ are symmetric, so strictly speaking it should not have a left-inverse. We discuss how to circumvent these issues in Section 6.3 for tensor ring decomposition and in the full version for low-rank factorization.

Expressing Kronecker structure of U. While most of the steps outlined in Sections 3.1 and 3.2 proving various properties of U are relatively straightforward to implement in SoS, e.g. (3) and (4) for tensor ring decomposition and rank preservation for low-rank factorization, it is less clear how to even express in SoS the main conclusion that we want to show about U, namely that it is the Kronecker power of an $r \times r$ orthogonal matrix V.

In particular, how do we express V? That is, how do we use the existing program variables to design a matrix V for which we could hope to prove U is its Kronecker power? For tensor ring decomposition, a natural candidate would be to take the $r \times r$ matrix U^{ii} for every $i \in [r]$, pick one of its nonzero columns and normalize it to a unit vector V^i, and take V's columns to consist of V^i's. This does not quite work because the normalization step involves a rational function of the entries of the program variables. To fix this, we need to carry around these normalization factors when expressing the U^{ij}'s as outer products.

While this turns out to be manageable for tensor ring decomposition, such an approach quickly becomes unwieldy for low-rank factorization where the degree ω can be arbitrary. Fortunately, for odd ω, there is a simpler workaround. Heuristically, because we expect to have $U^{i\cdots i} = V^i \otimes \cdots \otimes V^i$ for $r \times r$ orthogonal matrix V, we also expect that V^i is equal to the vector, call it $\widetilde{U}^i$, whose j-th entry is given by

$$\widetilde{U}^i_j \triangleq \sum_{j_1,\ldots,j_{\lfloor \omega/2 \rfloor} \in [r]} U^{i\cdots i}_{j_1 j_1 \cdots j_{\lfloor \omega/2 \rfloor} j_{\lfloor \omega/2 \rfloor} j}.$$

In particular, the entries of $\widetilde{U}$ are simply linear forms in those of U. For general odd ω, our SoS proof that U is a Kronecker power thus entails proving that U is the Kronecker power of $\widetilde{U}$ and that $\widetilde{U}$ is orthogonal.

Rounding by breaking gauge symmetry. Finally, we describe how to take a pseudodistribution satisfying the constraints of our SoS program and round to an integral solution. This is complicated by the fact that we can only hope to recover the ground truth up to gauge symmetry. We address this by breaking symmetry and imposing a small number of additional constraints to our SoS program. These constraints will ensure that the transformation U is not just the Kronecker power of some $r \times r$ orthogonal matrix, but actually equal to the identity matrix. This shows that Q_a and Q_a^* (or T_a and T_a^*) are not only equivalent up to rotation, but *equal*. At that point we can produce an integral solution simply by outputting the pseudoexpectations of $\{Q_a\}$ or $\{T_a\}$.

In tensor ring decomposition, a natural approach to ensure that U is identity would be to further insist that one of the Q_a's is diagonal with diagonal entries sorted in increasing order. The reason is that if the eigenvalues of Q_a are all distinct (more precisely, well-separated to account for noise when $\eta > 0$), then the only way for $VQ_aV^\top$ to be equal to Q_a for some $r \times r$ rotation V would be for V to be equal to $\pm\mathrm{Id}$ (and thus for U to be $(\pm\mathrm{Id})^{\otimes 2} = \mathrm{Id}$). This approach in fact already works in the smoothed setting.

To handle the slightly more general setting where $\{Q_a\}$ are "incoherent" but have repeated eigenvalues, we slightly modify this by insisting that some suitable *random linear combination* of the Q_a's is diagonal with sorted diagonal entries. By carefully designing how this linear combination is sampled we can ensure that it has sufficient eigengaps with high probability.

For low-rank factorization, we use a similar approach with various technical modifications to account for the fact that for $\omega > 2$, order-ω tensors do not have a suitable notion of eigengap. The details here are rather thorny and involve running *two* SoS relaxations in succession. We defer an overview of these workarounds to the full version.

Roadmap. In Section 4 we describe related work. After establishing the reduction from learning polynomial transformations to tensor ring decomposition and low-rank factorization in Section 5, we give a more detailed overview of our algorithm for tensor ring decomposition in Section 6, deferring many of the details of this, as well as all the details for low-rank factorization, to the full version.

input dimension	output dimension	degree	rank
r	d	ω	ℓ

Table 1: Notation for the main parameters of a polynomial transformation

4 RELATED WORK

There is a vast literature on density estimation of distributions, especially in high dimensions, to which we cannot do justice here. For conciseness we will only survey the most relevant work.

Learning latent variable models Much of the recent algorithmic success in high dimensional distribution learning has been in developing efficient algorithms for a variety of latent variable models, such as mixture models [1, 2, 4, 5, 12, 17, 18, 36, 37, 43, 44, 48, 50, 59, 63, 78, 79, 87, 88, 93, 100] and graphical models [10, 20–26, 32, 35, 40, 42, 45, 53, 61, 72, 77, 94, 102, 104]. Of these works, we highlight the work on learning latent variable models in smoothed settings [4, 5, 10, 18, 19, 50, 66], where a similar "blessing of dimensionality" phenomena to the one we observe can be seen.

However, there are important qualitative differences between these settings and the one we consider. While our model can be viewed as a latent variable model, where the hidden variable is the unknown Gaussian, the main challenge of our work is to learn the transformation of the hidden variable, rather than the hidden variable itself. This makes the problem take a qualitatively different form than much of the prior work. From a technical perspective, another difference between our setting and much of the prior work on learning latent variable models is that the form of the pdf for our distributions is much more implicit; in particular, the relationship between moments of the distribution and the pdf is much less clean than (say) for Gaussian mixture models.

(Non-linear) independent component analysis Independent component analysis as first proposed in [33] is the question of learning an (unknown) linear transformation of a non-Gaussian, coordinate-wise independent random variable. Here, the goal is to recover the underlying transformation as well as the original random variable (note that non-Gaussianity is necessary for this to be possible). The literature on ICA is incredibly large, so we refer

the reader to surveys of [34, 67, 69] and references within for a more detailed literature review. We briefly note that to our knowledge, one cannot black-box apply a kernelized version of the algorithms for ICA such as [9, 11, 49, 84, 98] to solve our problem, because in the polynomial kernel space, the resulting random variable does not satisfy coordinate-wise independence.

Of particular interest to us is the literature on non-linear ICA, which is very closely related to the learning problem we consider. However, in non-linear ICA, the goal is not just to learn a distribution which is close to the ground truth, but in fact to recover the original (i.e. pre-transformation) latent variables. Despite a substantial amount of interest in this model from the more applied side (see e.g. [68, 71, 74] and references therein), from a theoretical perspective, the problem remains relatively poorly understood without additional assumptions. it is known that in the worst case, the latent variable is not identifiable [70]. As another example of this phenomenon, note that the aforementioned counterexample of [57] from Section 2 implies that for cubic transformations of Gaussians, the latent variable is not always identifiable.

Consequently, much of the literature has shifted to consider data with temporal structure, see e.g. [68, 71]. In contrast, we consider the standard i.i.d. model, but we make stronger parametric assumptions about the transformation, namely, that it is a low-degree polynomial. In addition, we do not require that the latent variable be identifiable, as we only care about learning the underlying distribution, and not recovering the the latent variable.

Learning deep generative models A full literature on the theory of learning deep generative models, and GANs in particular, is beyond the scope of this paper. See e.g. [58] for a more in depth survey. In terms of end-to-end learning guarantees with efficient algorithms, the literature is somewhat sparser. To our knowledge, results are only known for relatively simple networks. Much of the literature focuses on understanding when stochastic first order methods can learn the distribution on toy generative models [3, 39, 47, 52, 73, 81]. One line of work considers the problem of learning distributions generated by pushforwards of Gaussians one-layer neural networks with ReLU activations [81, 103]. However, such distributions have a much simpler structure than the ones we consider in this paper, which correspond to two-layer neural networks (i.e. with one hidden layer). Indeed, when the neural networks only have one layer, this means that the output of the distribution is very similar to a truncated Gaussian, and one can leverage techniques from the literature of learning from truncated samples [38]. However, such structure completely disappears with two layer neural networks. In that sense, our guarantee is the first end-to-end provable result for learning pushforwards under neural networks beyond a single layer.

Arguably the closest paper to ours is the recent work of [83]. This paper considers a very similar setting to ours, however, their result has a number of drawbacks compared to ours. First, they assume that the hidden weight matrices are orthonormal; that is, the coordinates of their generative model are of the form $p(x) = \sum_{i=1}^{\ell} a_i \langle u_i, x\rangle^{\omega}$, where the u_i are orthogonal unit vectors. This is an incredibly brittle assumption, and their algorithm breaks even if the u_i have inverse polynomially small correlations. In particular, their assumption does not even hold in the smoothed setting we consider. In contrast, we handle arbitrary low-rank tensors. Second, their bounds scale exponentially with scale of a_i, whereas our bounds do not. Finally, their provable guarantees are contingent on a conjectured identifiability assumption which they do not prove (see discussion above Theorem 2 in [83]). Therefore, they do not give end-to-end provable guarantees for their learning task. In contrast, we give fully provable results for a significantly more general setting. Indeed, much of the technical work in our paper comes down to giving a proof of identifiability for a more involved tensor decomposition-style problem.

The relative lack of algorithms for these learning tasks may be inherent, at least in some worst case sense. Indeed, recent work of [29, 30] demonstrates that learning pushforwards of Gaussians under low-depth ReLU networks in Wasserstein distance is computationally intractable, either under standard cryptographic assumptions or in the statistical query model. The starting point for the former result is the observation that the assumption that "local pseudorandom generators" exist [6, 7, 54] implies that learning polynomial transformations of the uniform distribution over the hypercube is computationally intractable. The idea behind the result of [29] is that even depth-2 ReLU network pushforwards can match all low-order moments of a standard Gaussian distribution. Alongside our information theoretic lower bound against parameter estimation for polynomial pushforwards (see Appendix of full version), these hardness results give evidence that some sort of smoothing assumptions are necessary to make the problem algorithmically tractable.

On the flip side, there has been a lot of work on scrutinizing the ways in which the training dynamics for learning generative models in practice are aligned or misaligned with traditional statistical notions of distribution learning [8, 13, 46, 97], and relatedly, what it takes for minimax optimality (e.g. under the Wasserstein GAN objective) to actually ensure distribution learning [14, 27, 30, 85, 95, 96, 99]. While this suggests that a satisfactory theory for generative models may ultimately involve more than just distribution learning in the traditional sense, the basic algorithmic question considered in the present work, in addition to being natural in its own right, seems like a natural stepping stone towards such a theory.

Tensor ring decomposition Tensor ring decomposition is an important instance of tensor network decomposition and arises as a prototypical model for periodic one-dimensional physical systems [101]. As alluded to previously, the tensor ring format, along with other dimension-reduced tensor representations such as the tensor train format [90, 91], or those associated with Tucker rank or hierarchical Tucker rank [15, 89], arose as ways of representing large tensors implicitly. Unfortunately, unlike Tucker decomposition [41, 106], hierarchical Tucker decomposition [56], or tensor-train decomposition [91, 108], obtaining efficient algorithms with provable guarantees for tensor ring decomposition has proven quite challenging [31]. In part, this is because the notion of rank associated with tensor ring decomposition—in contrast to the other aforementioned representations—is unidentifiable in many scenarios [105]. While some heuristic algorithms for tensor ring decomposition have been proposed, such as those based on alternating least squares [75, 107], prior to our work, there were no known algorithms for the problem with end-to-end theoretical guarantees.

SoS for learning From a technical point of view, our algorithms fit into the recent SoS "proofs-to-algorithms" paradigm for statistical inference problems (see e.g. [62] for a more thorough overview). From a technical perpective, our problem is closest to the line of work using SoS and SoS-inspired algorithms to obtain efficient algorithms for a variety of tensor decomposition tasks [16, 51, 64, 65, 86]. However, our problem setting appears to be significantly more technically challenging, in large part because in addition to the usual permutational symmetry among components in tensor decomposition, there is an extra *gauge symmetry* inherent to the problems we consider. Even for tensor ring decomposition, which generalizes tensor decomposition, to our knowledge the techniques in these papers do not apply.

5 LEARNING POLYNOMIAL TRANSFORMATIONS

In this section, we establish the connection between the inverse problems of Section 2.1, tensor ring decomposition and low-rank factorization, to the problem of learning polynomial transformations:

Theorem 5.1. *Let $\varepsilon > 0$. Suppose there is an algorithm for tensor ring decomposition (Definition 2.6) that, given as input S, T satisfying*

$$\left|\operatorname{Tr}(Q_a^* Q_b^*) - S_{a,b}\right| \le \eta \ \ \forall\, a, b \in [d] \tag{6}$$

$$\left|\operatorname{Tr}(Q_a^* Q_b^* Q_c^*) - T_{a,b,c}\right| \le \eta \ \ \forall\, a, b, c \in [d]. \tag{7}$$

for some $\eta = \eta(\varepsilon)$, runs in time T and with high probability outputs symmetric matrices $\widehat{Q}_1, \dots, \widehat{Q}_d$ for which $d_{\mathrm{G}}(\{Q_a^*\}, \{\widehat{Q}_a\}) \le \varepsilon$.

Then there is an algorithm for parameter learning the transformation $\mathcal{D}$ given by quadratic network $Q_1^*, \dots, Q_d^*$ to error ε with high probability that draws $O(r^3\mathcal{R}^6 \log^3(2d/\delta)/\eta(\varepsilon)^2)$ samples and runs in time T. Furthermore, this algorithm also solves proper density estimation to Wasserstein error $O(\varepsilon r\sqrt{d})$ with high probability.

Theorem 5.2. *Let $\varepsilon > 0$ and define*

$$\Sigma \triangleq \mathop{\mathbb{E}}_{g \sim \mathcal{N}(0, \mathrm{Id}_r)} [g^{\otimes\omega} (g^{\otimes\omega})^\top]. \tag{8}$$

Suppose there is an algorithm for low-rank factorization (Definition 2.8) that, given as input S satisfying

$$\left|\langle T_a^*, T_b^* \rangle_\Sigma - S_{a,b}\right| \le \eta \ \ \forall\, a, b \in [d] \tag{9}$$

for some $\eta = \eta(\varepsilon)$, runs in time T and with high probability outputs $\widehat{T}_1, \dots, \widehat{T}_d$ for which we have $d_{\mathrm{G}}(\{T_a^*\}, \{\widehat{T}_a\}) \le \varepsilon$.

Then there is an algorithm for parameter learning the transformation $\mathcal{D}$ given by low-rank polynomial network $T_1^*, \dots, T_d^*$ to error ε with high probability that draws $O(\omega r)^{2\omega} \mathcal{R}^4 \log^{2\omega}(\delta/d)/\eta(\varepsilon)^2$ samples and runs in time T. Furthermore, this algorithm also solves proper density estimation to Wasserstein error $O(\varepsilon r\sqrt{d})$ with high probability.

5.1 Quadratic Transformations

Here we establish the connection between method of moments for learning quadratic transformations and tensor ring decomposition, and tensor ring decomposition and low-rank factorization. Throughout this section, let $\mathcal{D}$ be a d-dimensional degree-2 transformation with seed length r that is specified by the polynomial network $Q_1^*, \dots, Q_d^* \in \mathbb{R}^{r \times r}$.

Lemma 5.3. *If z is a sample from $\mathcal{D}$, then for any $a, b, c \in [d]$,*

$$2\operatorname{Tr}(Q_a^* Q_b^*) = \mathbb{E}[(z_a - \mathbb{E}[z_a])(z_b - \mathbb{E}[z_b])]$$

$$8\operatorname{Tr}(Q_a^* Q_b^* Q_c^*) = \mathbb{E}[(z_a - \mathbb{E}[z_a])(z_b - \mathbb{E}[z_b])(z_c - \mathbb{E}[z_c])]$$

Proof. For any $a, b, c \in [d]$, we have by Isserlis' theorem that

$$\mathbb{E}[z_a] = \mathop{\mathbb{E}}_{x \sim \mathcal{N}(0, \mathrm{Id})} [x^\top Q_a^* x] = \operatorname{Tr}(Q_a^*).$$

$$\mathbb{E}[z_a z_b] = \mathop{\mathbb{E}}_{x}[(x^\top Q_a^* x) \cdot (x^\top Q_b^* x)] = \operatorname{Tr}(Q_a^*)\operatorname{Tr}(Q_b^*) + 2\operatorname{Tr}(Q_a^* Q_b^*).$$

$$\begin{aligned}\mathbb{E}[z_a z_b z_c] &= \mathop{\mathbb{E}}_{x}[(x^\top Q_a^* x) \cdot (x^\top Q_b^* x) \cdot (x^\top Q_c^* x)] \\ &= \operatorname{Tr}(Q_a^*)\operatorname{Tr}(Q_b^*)\operatorname{Tr}(Q_c^*) + 2\operatorname{Tr}(Q_a^*)\operatorname{Tr}(Q_b^* Q_c^*) \\ &\quad + 2\operatorname{Tr}(Q_b^*)\operatorname{Tr}(Q_a^* Q_c^*) + 2\operatorname{Tr}(Q_c^*)\operatorname{Tr}(Q_a^* Q_b^*) \\ &\quad + 8\operatorname{Tr}(Q_a^* Q_b^* Q_c^*),\end{aligned}$$

where $x \sim \mathcal{N}(0, \mathrm{Id})$ and for the last identity, we used the fact that for any symmetric matrices A_1, A_2, A_3,

$$\operatorname{Tr}(A_1 A_2 A_3) = \operatorname{Tr}(A_{\pi(1)} A_{\pi(2)} A_{\pi(3)}) \ \ \forall\, \pi \in \mathcal{S}_3.$$

The lemma follows immediately from the above moment calculations. □

Lemma 5.4 (Empirical moment estimation)**.** *For any $\eta, \delta > 0$, if $\|Q_a^*\|_F^2 \le \mathcal{R}^2$ for all $a \in [d]$, there is an algorithm that takes $O(r^3\mathcal{R}^6 \log^3(2d/\delta)/\eta^2)$ samples from $\mathcal{D}$ and with probability at least $1 - \delta$ outputs $S \in \mathbb{R}^{d \times d}$ and $T \in \mathbb{R}^{d \times d \times d}$ satisfying (6) and (7).*

We defer the proof of this to the Appendix of the full version. Theorem 5.1 now immediately follows from Lemma 5.4.

General rotation-invariant seeds. While it would appear that the reduction above makes use of the special structure of Gaussian moments, our approach easily extends to any rotation-invariant seed distribution D which is reasonably concentrated so that the corresponding transformation moments can be estimated from samples as in Lemma 5.4. The reason for this comes from the following elementary observation about moments of rotation-invariant distributions, whose proof we defer to the full version

Lemma 5.5. *For any rotation-invariant distribution D over $\mathbb{R}^r$ and any degree-e homogeneous polynomial $q : \mathbb{R}^r \to \mathbb{R}$, $\mathbb{E}_{x \sim D}[q(x)] = C_{D,e} \cdot \mathbb{E}_{g \sim \mathcal{N}(0, \mathrm{Id})}[q(g)]$ for $C_{D,e} \triangleq \frac{\Gamma(e/2)}{2^e \cdot \Gamma((r+e)/2)} \cdot \mathbb{E}_{x \sim D}[\|x\|^e]$.*

So from the second-, fourth-, and sixth-order moments of any rotation-invariant D, we can extract the quantities $\operatorname{Tr}(Q_a^* Q_b^*)$ and $\operatorname{Tr}(Q_a^* Q_b^* Q_c^*)$ as in Lemma 5.3 even when D is not $\mathcal{N}(0, \mathrm{Id})$, provided we know $\mathbb{E}_{x \sim D}[\|x\|^e]$ for $e = 2, 4, 6$. Regarding this last point, we note that it is entirely reasonable to assume that these quantities, in fact even a description of D itself, is known to the algorithm designer: in the practice of generative models one has complete control over the seed distribution/prior that is used.

5.2 Low-Rank Transformations

Here we establish the connection between method of moments for learning low-rank transformations and low-rank factorization. Throughout this section, let $\mathcal{D}$ be a d-dimensional degree-ω transformation with seed length r that is specified by the low-rank polynomial network $T_1^*, \dots, T_d^* \in (\mathbb{R}^r)^{\otimes\omega}$.

Lemma 5.6. *If z is a sample from $\mathcal{D}$, then for any $a, b \in [d]$,*

$$\langle T_a^*, T_b^* \rangle_\Sigma = \mathbb{E}[z_a z_b],$$

where Σ is defined in (8).

Proof. This follows from

$$\begin{aligned}\mathbb{E}[z_a z_b] &= \mathop{\mathbb{E}}_{g \sim \mathcal{N}(0, \mathrm{Id}_r)} [\langle T_a^*, g^{\otimes \omega} \rangle \langle T_b^*, g^{\otimes \omega} \rangle] \\ &= \mathrm{vec}(T_a^*)\, \mathbb{E}[g^{\otimes \omega} (g^{\otimes \omega})^\top]\, \mathrm{vec}(T_b^*). \qquad \square\end{aligned}$$

Lemma 5.7 (Empirical moment estimation)**.** *For any $\eta, \delta > 0$, if $\|T_a^*\|_F^2 \le \mathcal{R}^2$ for all $a \in [d]$, there is an algorithm that takes $O(\omega r)^{2\omega} \mathcal{R}^4 \log^{2\omega}(\delta/d)/\eta^2$ samples from $\mathcal{D}$ and with probability at least $1 - \delta$ outputs $S \in \mathbb{R}^{d \times d}$ satisfying (9).*

We defer the proof of this to the Appendix of the full version. Theorem 5.2 now immediately follows from Lemma 5.7.

General rotation-invariant seeds. Note that Lemma 5.6 makes no use of the fact that the transformation has seed distribution given by $\mathcal{N}(0, \mathrm{Id})$, so our reduction from learning low-rank transformations to low-rank factorization easily carries over to any known seed distribution D which is sufficiently well-concentrated that the pairwise moments of $\mathcal{D}$ can be estimated from samples as in Lemma 5.7 and for which the corresponding low-rank factorization problem with Σ now given by $\mathbb{E}_{x \sim D}[\mathrm{vec}(x^{\otimes \omega})\, \mathrm{vec}(x^{\otimes \omega})^\top]$ is tractable. As we show in the full version, our algorithm for low-rank factorization applies to any Σ of this form for which the seed distribution D is *rotation-invariant* and for which very mild condition number bounds hold. In the full version, we also give an algorithm for low-rank factorization when $\Sigma = \mathrm{Id}$, which yields a learning algorithm for a certain family of *inhomogeneous* polynomial transformations given by one hidden layer networks with Hermite polynomial activations.

6 TENSOR RING DECOMPOSITION

Recall that in tensor ring decomposition (Definition 2.6), we are given $S \in \mathbb{R}^{d \times d}$ and $T \in \mathbb{R}^{d \times d \times d}$ such that there exist unknown symmetric matrices $Q_1^*, \dots, Q_d^* \in \mathbb{R}^{r \times r}$ satisfying

$$|S_{a,b} - \mathrm{Tr}(Q_a^* Q_b^*)| \le \eta \quad \text{and} \quad |T_{a,b,c} - \mathrm{Tr}(Q_a^* Q_b^* Q_c^*)| \le \eta \qquad (10)$$

for all $a, b, c \in [d]$. In this section we give a polynomial-time algorithm for recovering $Q_1^*, \dots, Q_d^*$ from S, T under the following assumptions:

Assumption 1. *For parameters $\mathcal{R} \ge 1, \kappa > 0$,*

1. *(Scaling) $\|Q_a^*\|_F \le \mathcal{R}$ for all $a \in [d]$.*
2. *(Condition number bound) $\sigma_{\binom{r+1}{2}}(M^*) \ge \kappa$, where $M^* \in \mathbb{R}^{d \times \binom{r+1}{2}}$ is the matrix whose $(a, (i_1, i_2))$-th entry, for $a \in [d]$ and $1 \le i_1 \le i_2 \le r$, is given by $(Q_a^*)_{i_1 i_2}$.*

Remark 6.1. *Readers familiar with the standard guarantees for Jennrich's algorithm will recognize that Part 2 of Assumption 1 is the tensor ring analogue of the condition number assumption in tensor decomposition. Namely, given an estimate of $\sum_i v_i^{\otimes 3}$, Jennrich's algorithm can recover $\{v_i\}$ provided the matrix whose columns consist of v_i is well-conditioned (see e.g. [18, Condition 2.2]).*[2]

One can readily check that Assumption 1 is gauge-invariant (see Appendix of full version for proof):

Lemma 6.2. *If $\{Q_a^*\}$ satisfy (10) and Assumption 1 with parameters $\mathcal{R}, \kappa$, then $\{V Q_a^* V^\top\}$ also satisfy (10) and Assumption 1 with the same parameters for any $V \in O(r)$.*

Under Assumption 1, we give an algorithm for tensor ring decomposition that runs in time polynomial in all parameters:

Theorem 6.3. *For $d \ge \binom{r+1}{2}$, suppose $Q_1^*, \dots, Q_d^* \in \mathbb{R}^{r \times r}$ satisfy Assumption 1 and $\eta \le O(\frac{\kappa^2}{r d^{3/2}})$, and we are given $S \in \mathbb{R}^{d \times d}$ and $T \in \mathbb{R}^{d \times d \times d}$ satisfying (10).*

Then there is an algorithm TensorRingDecompose*(S, T) which runs in time $\mathrm{poly}(d, r)$ and outputs $\widehat{Q}_1, \dots, \widehat{Q}_d$ for which we have the bound $d_{\mathrm{G}}(\{Q_a^*\}, \{\widehat{Q}_a\}) \le \mathrm{poly}(d, r, \mathcal{R}, 1/\kappa) \cdot \eta^c$ for some absolute constant $c > 0$, with high probability.*

Section overview. Our algorithm is based on rounding the solution to a suitable sum-of-squares relaxation. As such, our analysis is centered around exhibiting a low-degree sum-of-squares proof that the ground truth $\{Q_a^*\}$ is identifiable from S, T. As discussed in Section 3, the gauge symmetry inherent in tensor ring decomposition poses a major challenge for this, because $\{Q_a^*\}$ is only identifiable up to a global rotation in $\mathbb{R}^r$. In Section 6.1 we outline our strategy for "breaking symmetry" by imposing certain constraints on $\{Q_a^*\}$ that are without loss of generality but which will *uniquely* identify $\{Q_a^*\}$. In Section 6.2 we then formulate our sum-of-squares program which incorporates this symmetry-breaking strategy.

The high-level strategy will be to introduce SoS variables $\{Q_a\}$ that are constrained to have the same pairwise and three-wise moment bounds as in (10), and we would like to prove the $\{Q_a\}$ are close to $\{Q_a^*\}$ in Frobenius norm. To show this, we would like to show that the $r^2 \times r^2$ linear transformation that maps every $\mathrm{vec}(Q_a^*)$ to $\mathrm{vec}(Q_a)$ behaves like the Kronecker power $\mathrm{Id}_r^{\otimes 2}$. Because every Q_a^* and Q_a is symmetric, there is some ambiguity in formulating this transformation as an SoS variable (recall the discussion at the end of Section 3.1 of the technical overview).

In Section 6.3 we make a first attempt by constructing a certain auxiliary $r^2 \times r^2$ matrix variable U that, as we show in the full version, behaves in some respects like this $r^2 \times r^2$ transformation. The remaining details we defer to the full version. Namely, in the full version, we then use the third-order constraints in (10) to show that the entries of U satisfy a certain collection of quadratic relations. We use these quadratic relations to refine U to give another SoS auxiliary variable W which better captures the $r^2 \times r^2$ transformation and which also satisfies a similar collection of quadratic relations as U. We then complete the analysis by implementing the aforementioned symmetry-breaking strategy in SoS to show that W is approximately $\mathrm{Id}_r^{\otimes 2}$. Finally, in the full version we use this to give our main algorithm TensorRingDecompose and prove Theorem 6.3. We also show how to improve the runtime of Theorem 6.3 to only depend *linearly* on d.

[2] Technically if $\{Q_a^*\}$ are all diagonal with $(Q_a^*)_{ii} = (v_i)_a$, Part 2 of Assumption 1 does not apply because M^* will have many zero entries, but it is straightforward to modify our sum-of-squares algorithm to incorporate the assumption that $\{Q_a^*\}$ are diagonal to recover the guarantees of Jennrich's algorithm.

6.1 Breaking Gauge Symmetry for the Ground Truth

A natural approach for breaking symmetry would be to insist without loss of generality that, for instance, Q_1^* is diagonal with sorted entries. If the eigenvalues of Q_1^* are well-separated, then one can check that the only rotations $V \in O(r)$ for which $V^\top Q_a^* V = Q_a^*$ for all $a \in [d]$ are those for which V is diagonal with diagonal entries in $\{\pm 1\}$. If we could additionally insist that, say, the first row of Q_2^* consisted entirely of *strictly positive* entries, this would force V to be the identity and completely break the gauge symmetry.

Of course, it could be that Q_1^* and Q_2^* don't meet the desired criteria for making such assumptions: Q_1^* might have some repeated eigenvalues, or Q_2^* might have a zero entry in its first row.[3] But the above strategy is certainly not specific to Q_1^* or Q_2^* or the choice of row in Q_2^*. Indeed, it would be enough for this to hold for some fixed linear combinations of $\{Q_a^*\}$, instead of for Q_1^*, and Q_2^* respectively.

We show that under Assumption 1, there is indeed a way to construct such linear combinations. In the Appendix in the full version, we give an algorithm that takes in S and outputs linear combinations of $\{Q_a^*\}$ satisfying the desired properties, which we formalize in the definition below:

Definition 6.4. *We say that $\lambda, \mu \in \mathbb{S}^{d-1}$ are υ-*non-degenerate combinations of $Q_1^*, \ldots, Q_d^*$ *if the following two properties hold for*

$$Q_\lambda^* \triangleq \sum_{a\in[d]} \lambda_a Q_a^* \qquad \text{and} \qquad Q_\mu^* \triangleq \sum_{a\in[d]} \mu_a Q_a^*. \tag{11}$$

(1) Q_λ^* has minimum eigengap at least υ.
(2) Let $V^\top \Lambda V$ be the eigendecomposition of Q_μ^*. Then every entry of $V Q_\mu^* V^\top$ has magnitude at least υ.

Because Assumption 1 is gauge-invariant by Lemma 6.2, we can assume without loss of generality that Q_λ^* defined in (11) is diagonal with entries sorted in nondecreasing order. As Q_λ^* has minimum eigengap at least υ,

$$(Q_\lambda^*)_{jj} \geq (Q_\lambda^*)_{ii} + \upsilon \;\; \forall\, j > i. \tag{12}$$

After diagonalizing Q_λ^*, the second part of Definition 6.4 implies that $|(Q_\mu^*)_{ij}| \geq \upsilon$ for all $i, j \in [r]$.

By applying one more joint rotation to $Q_1^*, \ldots, Q_d^*$ given by a diagonal matrix of ± 1 entries, we can additionally assume that the first row of Q_μ^* consists of *nonnegative* entries. That is,

$$(Q_\mu^*)_{1j} \geq \upsilon \;\; \forall\, j \in [r]. \tag{13}$$

In the sequel, we will show how to recover $Q_1^*, \ldots, Q_d^*$ in Frobenius norm (as opposed to just parameter distance) by insisting that our estimates also satisfy (12) and (13).

6.2 A Sum-of-Squares Relaxation

To prove Theorem 6.3, we will use the following sum-of-squares program:

Program 1. *(Tensor Ring Decomposition)*

Parameters: *$\lambda, \mu \in \mathbb{S}^{d-1}$, $S \in \mathbb{R}^{d\times d}$, $T \in \mathbb{R}^{d\times d\times d}$, $\mathcal{R} \geq 1$, $\kappa, \upsilon > 0$.*
Variables: *Let $Q_1, \ldots, Q_d$ be $r \times r$ matrix-valued variables, and let L be an $\binom{r+1}{2} \times d$ matrix-valued variable. Let M be the $d \times \binom{r+1}{2}$ matrix of indeterminates whose $(a, (i_1, i_2))$-th entry, for $a \in [d]$ and $1 \leq i_1 \leq i_2 \leq r$, is given by $(Q_a)_{i_1 i_2}$. Also define $Q_\lambda \triangleq \sum_{a=1}^d \lambda_a Q_a$ and $Q_\mu \triangleq \sum_{a=1}^d \mu_a Q_a$.*
Constraints:
(1) (Symmetry): $Q_a = Q_a^\top$ for all $a \in [d]$.
(2) (Second moments match): $-\eta \leq \mathrm{Tr}(Q_a Q_b) - S_{a,b} \leq \eta$ for all $a, b \in [d]$.
(3) (Third moments match): $-\eta \leq \mathrm{Tr}(Q_a Q_b Q_c) - T_{a,b,c} \leq \eta$ for all $a, b, c \in [d]$.
(4) (Q's bounded): $\|Q_a\|_F^2 \leq \mathcal{R}^2$ for all $a \in [d]$.
(5) (Left-inverse L): $LM = \mathrm{Id}$
(6) (L bounded): $\|L\|_F^2 \leq r^2/\kappa^2$.
(7) (Q_λ diagonal): $(Q_\lambda)_{ij} = 0$ for all $i \neq j$.
(8) (Q_λ sorted): $(Q_\lambda)_{jj} \geq (Q_\lambda)_{ii}$ for all $j > i$.
(9) (Q_μ's first row): $(Q_\mu)_{1j} \geq 0$ for all $j \in [r]$.

We can easily verify that the ground truth is feasible.

Lemma 6.5. *When $d \geq \binom{r+1}{2}$, the pseudodistribution given by the point distribution supported on $(Q_1^*, \ldots, Q_d^*, L^*)$, where L^* is the left inverse of M^*, is a feasible solution to Program 1.*

Proof. Note that L^* is well-defined by Part 2 of Assumption 1. It is immediate that Constraints 1-5 are satisfied, and Constraints 7-9 are satisfied by (12) and (13). For Constraint 6, note that $\|L^*\|_{\mathrm{op}} \leq 1/\kappa$ by Part 2 of Assumption 1, so $\|L^*\|_F^2 \leq \binom{r+1}{2}/\kappa^2 \leq r^2/\kappa^2$. □

The main result we will show about this sum-of-squares program is the following:

Theorem 6.6. *Suppose Assumption 1 holds, and for any $\lambda, \mu \in \mathbb{S}^{d-1}$ let $\widetilde{\mathbb{E}}[\cdot]$ be a degree-96 pseudo-expectation over the variables $Q_1, \ldots, Q_d, L$ satisfying the constraints of Program 1.*[4]

Then if λ, μ are υ-non-degenerate combinations of $Q_1^*, \ldots, Q_d^*$ for some $\upsilon > 0$, then $\|\widetilde{\mathbb{E}}[Q_a] - Q_a^*\|_F \leq \mathrm{poly}(d, r, \mathcal{R}, 1/\kappa, 1/\upsilon) \cdot \eta^c$ for all $a \in [d]$ for some absolute constant $c > 0$.

6.3 Hidden Rotation Variable

In this section we use the SoS variables of Program 1 to design an auxiliary "rotation variable" U that will play the role of the unknown linear transformation sending every Q_a^* to Q_a, after which the focus of our analysis in subsequent sections will be to show this transformation qualitatively behaves like $\mathrm{Id}_r^{\otimes 2}$.

First, define the $d \times r^2$ matrix N^* (resp. N) to be the matrix whose $(a, (i_1, i_2))$-th entry is given by $(Q_a^*)_{i_1 i_2}$ (resp. $(Q_a)_{i_1 i_2}$) for all $a \in [d]$, $i, j \in [r]$. Note that M, M^* are submatrices of N, N^*. Because $\mathrm{Tr}(Q_a^* Q_b^*) = (N^* N^{*\top})_{ab}$ and $\mathrm{Tr}(Q_a Q_b) = (N N^\top)_{ab}$, the first part of Eq. (10) and Constraint 2 imply that $\|N N^\top - N^* N^{*\top}\|_{\max} \leq \eta$.

A natural way to encode the unknown linear transformation from Q_a^* to Q_a as an auxiliary variable would be to consider something like $N^{-1} N^*$, because $(N^{-1} N^*) N^{*\top} \approx N^\top$, and the a-th column of this approximate equality between matrices implies that

[3] When Q_1^*, Q_2^* are smoothed, this will not happen, but in this section we opt for an algorithm that can work under minimal non-degeneracy assumptions even when $\{Q_a^*\}$ are not smoothed.

[4] We made no effort to optimize the degree of our SoS proof and suspect that with a little more work, this constant can be made much smaller.

the transformation $N^{-1}N^*$ maps Q_a^* to Q_a. By right multiplying this approximate equality by $(N^{-1})^\top$, we also see that $N^{-1}N^*$ is approximately orthogonal.

Of course, strictly speaking such a construction isn't well-defined: N is an SoS variable, so there is no meaningful notion of a left inverse N^{-1}. In fact there isn't even a suitable left inverse for the *scalar* matrix N^*, as N^* has duplicate columns (because every Q_a^* is symmetric). Nevertheless, we will use L as a proxy for N^{-1} and, with a few modifications, our construction of the "rotation variable" U will behave like $N^{-1}N^*$.

Formally, to construct U, first define the $\binom{r+1}{2} \times r^2$ matrix $\widehat{U}$ by

$$\widehat{U} \triangleq LN^*.$$

Then define the $r^2 \times r^2$ matrix U as follows. For any $i_1, i_2 \in [r]$, the (i_1, i_2)-th row of U is given by

$$U_{i_1 i_2} = \begin{cases} \widehat{U}_{i_1 i_2} & \text{if } i_1 = i_2 \\ \frac{1}{2}\widehat{U}_{i_1 i_2} & \text{if } i_1 < i_2 \\ \frac{1}{2}\widehat{U}_{i_2 i_1} & \text{if } i_1 > i_2 \end{cases}$$

When the context is clear, we will refer to $\mathrm{mat}(U_{i_1 i_2})$ as simply $U_{i_1 i_2}$, and similarly for any $j_1, j_2 \in [r]$, we will refer to $\mathrm{mat}(U^{j_1 j_2})$ as simply $U^{j_1 j_2}$. Note that the entries of U are (unknown) linear forms in the indeterminate entries of L.

Acknowledgments. The authors would like to thank Sebastien Bubeck and Raghu Meka for illuminating discussions in the early stages of this work. Yuanzhi Li is supported by NSF-Career 2145703.

REFERENCES

[1] Jayadev Acharya, Ashkan Jafarpour, Alon Orlitsky, and Ananda Theertha Suresh. 2014. Near-optimal-sample estimators for spherical gaussian mixtures. *arXiv preprint arXiv:1402.4746* (2014).

[2] Dimitris Achlioptas and Frank McSherry. 2005. On spectral learning of mixtures of distributions. In *International Conference on Computational Learning Theory*. Springer, 458–469.

[3] Zeyuan Allen-Zhu and Yuanzhi Li. 2021. Forward Super-Resolution: How Can GANs Learn Hierarchical Generative Models for Real-World Distributions. *arXiv preprint arXiv:2106.02619* (2021).

[4] Animashree Anandkumar, Rong Ge, Daniel Hsu, Sham M Kakade, and Matus Telgarsky. 2014. Tensor decompositions for learning latent variable models. *Journal of machine learning research* 15 (2014), 2773–2832.

[5] Joseph Anderson, Mikhail Belkin, Navin Goyal, Luis Rademacher, and James Voss. 2014. The more, the merrier: the blessing of dimensionality for learning large Gaussian mixtures. In *Conference on Learning Theory*. PMLR, 1135–1164.

[6] Benny Applebaum. 2016. Cryptographic hardness of random local functions. *Computational complexity* 25, 3 (2016), 667–722.

[7] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. 2006. Cryptography in NC^0. *SIAM J. Comput.* 36, 4 (2006), 845–888.

[8] Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. 2017. Generalization and equilibrium in generative adversarial nets (gans). In *International Conference on Machine Learning*. PMLR, 224–232.

[9] Sanjeev Arora, Rong Ge, Tengyu Ma, and Ankur Moitra. 2015. Simple, efficient, and neural algorithms for sparse coding. In *Conference on learning theory*. PMLR, 113–149.

[10] Sanjeev Arora, Rong Ge, Tengyu Ma, and Andrej Risteski. 2017. Provable learning of noisy-or networks. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*. 1057–1066.

[11] Sanjeev Arora, Rong Ge, Ankur Moitra, and Sushant Sachdeva. 2012. Provable ICA with unknown Gaussian noise, with implications for Gaussian mixtures and autoencoders. *Advances in Neural Information Processing Systems* 25 (2012).

[12] Sanjeev Arora and Ravi Kannan. 2005. Learning mixtures of separated nonspherical Gaussians. *The Annals of Applied Probability* 15, 1A (2005), 69–92.

[13] Sanjeev Arora, Andrej Risteski, and Yi Zhang. 2018. Do GANs learn the distribution? some theory and empirics. In *International Conference on Learning Representations*.

[14] Yu Bai, Tengyu Ma, and Andrej Risteski. 2018. Approximability of Discriminators Implies Diversity in GANs. In *International Conference on Learning Representations*.

[15] Jonas Ballani, Lars Grasedyck, and Melanie Kluge. 2013. Black box approximation of tensors in hierarchical Tucker format. *Linear algebra and its applications* 438, 2 (2013), 639–657.

[16] Boaz Barak, Jonathan A Kelner, and David Steurer. 2015. Dictionary learning and tensor decomposition via the sum-of-squares method. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*. 143–151.

[17] Mikhail Belkin and Kaushik Sinha. 2015. Polynomial learning of distribution families. *SIAM J. Comput.* 44, 4 (2015), 889–911.

[18] Aditya Bhaskara, Moses Charikar, Ankur Moitra, and Aravindan Vijayaraghavan. 2014. Smoothed analysis of tensor decompositions. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*. 594–603.

[19] Aditya Bhaskara, Aidao Chen, Aidan Perreault, and Aravindan Vijayaraghavan. 2019. Smoothed analysis in unsupervised learning via decoupling. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 582–610.

[20] Arnab Bhattacharyya, Sutanu Gayen, Kuldeep S Meel, and NV Vinodchandran. 2020. Efficient distance approximation for structured high-dimensional distributions via learning. *Advances in Neural Information Processing Systems* 33 (2020), 14699–14711.

[21] Arnab Bhattacharyya, Sutanu Gayen, Eric Price, and NV Vinodchandran. 2021. Near-optimal learning of tree-structured distributions by Chow-Liu. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*. 147–160.

[22] Enric Boix-Adsera, Guy Bresler, and Frederic Koehler. 2021. Chow-liu++: Optimal prediction-centric learning of tree ising models. *arXiv preprint arXiv:2106.03969* (2021).

[23] Guy Bresler. 2015. Efficiently learning Ising models on arbitrary graphs. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*. 771–782.

[24] Guy Bresler and Mina Karzand. 2020. Learning a tree-structured Ising model in order to make predictions. *The Annals of Statistics* 48, 2 (2020), 713–737.

[25] Guy Bresler, Elchanan Mossel, and Allan Sly. 2008. Reconstruction of Markov random fields from samples: Some observations and algorithms. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*. Springer, 343–356.

[26] Johannes Brustle, Yang Cai, and Constantinos Daskalakis. 2020. Multi-item mechanisms without item-independence: Learnability via robustness. In *Proceedings of the 21st ACM Conference on Economics and Computation*. 715–761.

[27] Minshuo Chen, Wenjing Liao, Hongyuan Zha, and Tuo Zhao. 2020. Statistical guarantees of generative adversarial networks for distribution estimation. *arXiv preprint arXiv:2002.03938* (2020).

[28] Sitan Chen, Aravind Gollakota, Adam R. Klivans, and Raghu Meka. 2022. Hardness of noise-free learning for two-hidden-layer neural networks. In *NeurIPS*.

[29] Sitan Chen, Jerry Li, and Yuanzhi Li. 2022. Learning (very) simple generative models is hard. In *NeurIPS*.

[30] Sitan Chen, Jerry Li, Yuanzhi Li, and Raghu Meka. 2022. Minimax Optimality (Probably) Doesn't Imply Distribution Learning for GANs. *arXiv preprint arXiv:2201.07206* (2022).

[31] Ziang Chen, Yingzhou Li, and Jianfeng Lu. 2020. Tensor ring decomposition: optimization landscape and one-loop convergence of alternating least squares. *SIAM J. Matrix Anal. Appl.* 41, 3 (2020), 1416–1442.

[32] CKCN Chow and Cong Liu. 1968. Approximating discrete probability distributions with dependence trees. *IEEE transactions on Information Theory* 14, 3 (1968), 462–467.

[33] Pierre Comon. 1994. Independent component analysis, a new concept? *Signal processing* 36, 3 (1994), 287–314.

[34] Pierre Comon and Christian Jutten. 2010. *Handbook of Blind Source Separation: Independent component analysis and applications*. Academic press.

[35] Sanjoy Dasgupta. 1997. The sample complexity of learning fixed-structure Bayesian networks. *Machine Learning* 29, 2 (1997), 165–180.

[36] Sanjoy Dasgupta. 1999. Learning mixtures of Gaussians. In *40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039)*. IEEE, 634–644.

[37] Sanjoy Dasgupta and Leonard J Schulman. 2007. A probabilistic analysis of EM for mixtures of separated, spherical Gaussians. *Journal of Machine Learning Research* 8 (2007), 203–226.

[38] Constantinos Daskalakis, Themis Gouleakis, Chistos Tzamos, and Manolis Zampetakis. 2018. Efficient statistics, in high dimensions, from truncated samples. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 639–649.

[39] Constantinos Daskalakis, Andrew Ilyas, Vasilis Syrgkanis, and Haoyang Zeng. 2017. Training gans with optimism. *arXiv preprint arXiv:1711.00141* (2017).

[40] Constantinos Daskalakis and Qinxuan Pan. 2021. Sample-optimal and efficient learning of tree Ising models. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*. 133–146.

[41] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. 2000. On the best rank-1 and rank-$(r_1, r_2, ..., r_n)$ approximation of higher-order tensors. *SIAM journal on Matrix Analysis and Applications* 21, 4 (2000), 1324–1342.

[42] Luc Devroye, Abbas Mehrabian, and Tommy Reddad. 2020. The minimax learning rates of normal and Ising undirected graphical models. *Electronic Journal of Statistics* 14, 1 (2020), 2338–2361.

[43] Ilias Diakonikolas and Daniel M Kane. 2020. Small covers for near-zero sets of polynomials and learning latent variable models. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 184–195.

[44] Ilias Diakonikolas, Daniel M Kane, and Alistair Stewart. 2018. List-decodable robust mean estimation and learning mixtures of spherical Gaussians. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*. 1047–1060.

[45] Ilias Diakonikolas, Daniel M Kane, Alistair Stewart, and Yuxin Sun. 2021. Outlier-robust learning of ising models under dobrushin's condition. In *Conference on Learning Theory*. PMLR, 1645–1682.

[46] William Fedus, Mihaela Rosca, Balaji Lakshminarayanan, Andrew M Dai, Shakir Mohamed, and Ian Goodfellow. 2017. Many paths to equilibrium: GANs do not need to decrease a divergence at every step. *arXiv preprint arXiv:1710.08446* (2017).

[47] Soheil Feizi, Farzan Farnia, Tony Ginart, and David Tse. 2017. Understanding gans: the lqg setting. *arXiv preprint arXiv:1710.10793* (2017).

[48] Jon Feldman, Ryan O'Donnell, and Rocco A Servedio. 2008. Learning mixtures of product distributions over discrete domains. *SIAM J. Comput.* 37, 5 (2008), 1536–1564.

[49] Alan Frieze, Mark Jerrum, and Ravi Kannan. 1996. Learning linear transformations. In *Proceedings of 37th Conference on Foundations of Computer Science*. IEEE, 359–368.

[50] Rong Ge, Qingqing Huang, and Sham M Kakade. 2015. Learning mixtures of gaussians in high dimensions. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*. 761–770.

[51] Rong Ge and Tengyu Ma. 2015. Decomposing overcomplete 3rd order tensors using sum-of-squares algorithms. *arXiv preprint arXiv:1504.05287* (2015).

[52] Gauthier Gidel, Reyhane Askari Hemmat, Mohammad Pezeshki, Rémi Le Priol, Gabriel Huang, Simon Lacoste-Julien, and Ioannis Mitliagkas. 2019. Negative momentum for improved game dynamics. In *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 1802–1811.

[53] Surbhi Goel. 2020. Learning ising and potts models with latent variables. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 3557–3566.

[54] Oded Goldreich. 2011. Candidate one-way functions based on expander graphs. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*. Springer, 76–87.

[55] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems* 27 (2014).

[56] Lars Grasedyck. 2010. Hierarchical singular value decomposition of tensors. *SIAM journal on matrix analysis and applications* 31, 4 (2010), 2029–2054.

[57] F Alberto Grünbaum. 1975. Cubic forms in Gaussian variables. *Illinois Journal of Mathematics* 19, 3 (1975), 405–411.

[58] Jie Gui, Zhenan Sun, Yonggang Wen, Dacheng Tao, and Jieping Ye. 2021. A review on generative adversarial networks: Algorithms, theory, and applications. *IEEE Transactions on Knowledge and Data Engineering* (2021).

[59] Moritz Hardt and Eric Price. 2015. Tight bounds for learning a mixture of two gaussians. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*. 753–760.

[60] RA HARSHMAN. 1970. Foundations of the PARAFAC procedure: Models and conditions for an" explanatory" multi-mode factor analysis. *UCLA Working Papers in Phonetics* 16 (1970), 1–84.

[61] Klaus-U Höffgen. 1993. Learning and robust learning of product distributions. In *Proceedings of the sixth annual conference on Computational learning theory*. 77–83.

[62] Samuel Hopkins. 2018. *Statistical inference and the sum of squares method*. Ph. D. Dissertation. Cornell University.

[63] Samuel B Hopkins and Jerry Li. 2018. Mixture models, robustness, and sum of squares proofs. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*. 1021–1034.

[64] Samuel B Hopkins, Tselil Schramm, and Jonathan Shi. 2019. A robust spectral algorithm for overcomplete tensor decomposition. In *Conference on Learning Theory*. PMLR, 1683–1722.

[65] Samuel B Hopkins, Tselil Schramm, Jonathan Shi, and David Steurer. 2016. Fast spectral algorithms from sum-of-squares proofs: tensor decomposition and planted sparse vectors. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*. 178–191.

[66] Daniel Hsu and Sham M Kakade. 2013. Learning mixtures of spherical gaussians: moment methods and spectral decompositions. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*. 11–20.

[67] Aapo Hyvärinen, Juha Karhunen, and Erkki Oja. 2002. Independent component analysis. *Studies in informatics and control* 11, 2 (2002), 205–207.

[68] Aapo Hyvarinen and Hiroshi Morioka. 2016. Unsupervised feature extraction by time-contrastive learning and nonlinear ica. *Advances in Neural Information Processing Systems* 29 (2016).

[69] Aapo Hyvärinen and Erkki Oja. 2000. Independent component analysis: algorithms and applications. *Neural networks* 13, 4-5 (2000), 411–430.

[70] Aapo Hyvärinen and Petteri Pajunen. 1999. Nonlinear independent component analysis: Existence and uniqueness results. *Neural networks* 12, 3 (1999), 429–439.

[71] Aapo Hyvarinen, Hiroaki Sasaki, and Richard Turner. 2019. Nonlinear ICA using auxiliary variables and generalized contrastive learning. In *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 859–868.

[72] Vishesh Jain, Frederic Koehler, and Andrej Risteski. 2019. Mean-field approximation, convex hierarchies, and the optimality of correlation rounding: a unified perspective. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. 1226–1236.

[73] Samy Jelassi, Arthur Mensch, Gauthier Gidel, and Yuanzhi Li. 2022. Adam is no better than normalized SGD: Dissecting how adaptivity improves GAN performance. https://openreview.net/forum?id=D9SuLzhgK9

[74] Ilyes Khemakhem, Diederik Kingma, Ricardo Monti, and Aapo Hyvarinen. 2020. Variational autoencoders and nonlinear ica: A unifying framework. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2207–2217.

[75] Yuehaw Khoo, Jianfeng Lu, and Lexing Ying. 2021. Efficient Construction of Tensor Ring Representations from Sampling. *Multiscale Modeling & Simulation* 19, 3 (2021), 1261–1284.

[76] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[77] Adam Klivans and Raghu Meka. 2017. Learning graphical models using multiplicative weights. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 343–354.

[78] Pravesh K Kothari, Jacob Steinhardt, and David Steurer. 2018. Robust moment estimation and improved clustering via sum of squares. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*. 1035–1046.

[79] Amit Kumar and Ravindran Kannan. 2010. Clustering with spectral norm and the k-means algorithm. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE, 299–308.

[80] Joseph M Landsberg and Giorgio Ottaviani. 2013. Equations for secant varieties of Veronese and other varieties. *Annali di Matematica Pura ed Applicata* 192, 4 (2013), 569–606.

[81] Qi Lei, Jason Lee, Alex Dimakis, and Constantinos Daskalakis. 2020. SGD learns one-layer networks in wgans. In *International Conference on Machine Learning*. PMLR, 5799–5808.

[82] Sue E Leurgans, Robert T Ross, and Rebecca B Abel. 1993. A decomposition for three-way arrays. *SIAM J. Matrix Anal. Appl.* 14, 4 (1993), 1064–1083.

[83] Yuanzhi Li and Zehao Dou. 2020. Making Method of Moments Great Again?–How can GANs learn distributions. *arXiv preprint arXiv:2003.04033* (2020).

[84] Yuanzhi Li and Yingyu Liang. 2017. Provable alternating gradient descent for non-negative matrix factorization with strong correlations. In *International Conference on Machine Learning*. PMLR, 2062–2070.

[85] Tengyuan Liang. 2018. How well generative adversarial networks learn distributions. *arXiv preprint arXiv:1811.03179* (2018).

[86] Tengyu Ma, Jonathan Shi, and David Steurer. 2016. Polynomial-time tensor decompositions with sum-of-squares. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 438–446.

[87] Dustin G Mixon, Soledad Villar, and Rachel Ward. 2017. Clustering subgaussian mixtures by semidefinite programming. *Information and Inference: A Journal of the IMA* 6, 4 (2017), 389–415.

[88] Ankur Moitra and Gregory Valiant. 2010. Settling the polynomial learnability of mixtures of gaussians. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE, 93–102.

[89] Alexander Novikov, Anton Rodomanov, Anton Osokin, and Dmitry Vetrov. 2014. Putting MRFs on a tensor train. In *International Conference on Machine Learning*. PMLR, 811–819.

[90] Ivan Oseledets and Eugene Tyrtyshnikov. 2010. TT-cross approximation for multidimensional arrays. *Linear Algebra Appl.* 432, 1 (2010), 70–88.

[91] Ivan V Oseledets. 2011. Tensor-train decomposition. *SIAM Journal on Scientific Computing* 33, 5 (2011), 2295–2317.

[92] Stanley Osher, Zuoqiang Shi, and Wei Zhu. 2017. Low dimensional manifold model for image processing. *SIAM Journal on Imaging Sciences* 10, 4 (2017), 1669–1690.

[93] Oded Regev and Aravindan Vijayaraghavan. 2017. On learning mixtures of well-separated gaussians. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 85–96.

[94] Andrej Risteski. 2016. How to calculate partition functions using convex programming hierarchies: provable bounds for variational methods. In *Conference on Learning Theory*. PMLR, 1402–1416.

[95] Nicolas Schreuder, Victor-Emmanuel Brunel, and Arnak Dalalyan. 2021. Statistical guarantees for generative models without domination. In *Algorithmic Learning Theory*. PMLR, 1051–1071.

[96] Shashank Singh, Ananya Uppal, Boyue Li, Chun-Liang Li, Manzil Zaheer, and Barnabás Póczos. 2018. Nonparametric Density Estimation under Adversarial

Losses. In *NeurIPS*.

[97] Jan Stanczuk, Christian Etmann, Lisa Maria Kreusser, and Carola-Bibiane Schönlieb. 2021. Wasserstein GANs work because they fail (to approximate the Wasserstein distance). *arXiv preprint arXiv:2103.01678* (2021).

[98] Ju Sun, Qing Qu, and John Wright. 2016. Complete dictionary recovery over the sphere I: Overview and the geometric picture. *IEEE Transactions on Information Theory* 63, 2 (2016), 853–884.

[99] Ananya Uppal, Shashank Singh, and Barnabas Poczos. 2019. Nonparametric Density Estimation & Convergence Rates for GANs under Besov IPM Losses. *Advances in Neural Information Processing Systems* 32 (2019), 9089–9100.

[100] Santosh Vempala and Grant Wang. 2004. A spectral algorithm for learning mixture models. *J. Comput. System Sci.* 68, 4 (2004), 841–860.

[101] Frank Verstraete, Diego Porras, and J Ignacio Cirac. 2004. Density matrix renormalization group and periodic boundary conditions: A quantum information perspective. *Physical review letters* 93, 22 (2004), 227205.

[102] Rui Wu, R Srikant, and Jian Ni. 2013. Learning loosely connected Markov random fields. *Stochastic Systems* 3, 2 (2013), 362–404.

[103] Shanshan Wu, Alexandros G Dimakis, and Sujay Sanghavi. 2019. Learning distributions generated by one-layer ReLU networks. *Advances in neural information processing systems* 32 (2019).

[104] Shanshan Wu, Sujay Sanghavi, and Alexandros G Dimakis. 2019. Sparse logistic regression learns all discrete pairwise graphical models. *Advances in Neural Information Processing Systems* 32 (2019).

[105] Ke Ye and Lek-Heng Lim. 2018. Tensor network ranks. *arXiv preprint arXiv:1801.02662* (2018).

[106] Anru Zhang and Dong Xia. 2018. Tensor SVD: Statistical and computational limits. *IEEE Transactions on Information Theory* 64, 11 (2018), 7311–7338.

[107] Qibin Zhao, Guoxu Zhou, Shengli Xie, Liqing Zhang, and Andrzej Cichocki. 2016. Tensor ring decomposition. *arXiv preprint arXiv:1606.05535* (2016).

[108] Yuchen Zhou, Anru R Zhang, Lili Zheng, and Yazhen Wang. 2022. Optimal high-order tensor svd via tensor-train orthogonal iteration. *IEEE Transactions on Information Theory* (2022).