On Finite-Sample Identifiability of Contrastive Learning-Based Nonlinear Independent Component Analysis

Qi Lyu [1] **Xiao Fu** [1]

Abstract

Nonlinear independent component analysis (nICA) aims at recovering statistically independent latent components that are mixed by unknown nonlinear functions. Central to nICA is the identifiability of the latent components, which had been elusive until very recently. Specifically, Hyvärinen *et al.* have shown that the nonlinearly mixed latent components are identifiable (up to often inconsequential ambiguities) under a generalized contrastive learning (GCL) formulation, given that the latent components are independent conditioned on a certain auxiliary variable. The GCL-based identifiability of nICA is elegant, and establishes interesting connections between nICA and popular unsupervised/self-supervised learning paradigms in representation learning, causal learning, and factor disentanglement. However, existing identifiability analyses of nICA all build upon an unlimited sample assumption and the use of ideal universal function learners—which creates a non-negligible gap between theory and practice. Closing the gap is a nontrivial challenge, as there is a lack of established "textbook" routine for finite sample analysis of such unsupervised problems. This work puts forth a finite-sample identifiability analysis of GCL-based nICA. Our analytical framework judiciously combines the properties of the GCL loss function, statistical generalization analysis, and numerical differentiation. Our framework also takes the learning function's approximation error into consideration, and reveals an intuitive trade-off between the complexity and expressiveness of the employed function learner. Numerical experiments are used to validate the theorems.

[1]School of EECS, Oregon State University, Corvallis, OR, United States. Correspondence to: Xiao Fu <xiao.fu@oregonstate.edu>, Qi Lyu <lyuqi@oregonstate.edu>.

Proceedings of the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

1. Introduction

Independent component analysis (ICA) has been an indispensable unsupervised learning tool across multiple domains. Theory and methods have been developed for ICA since the 1990s; see, e.g., (Comon, 1994). The classic ICA guarantees to identify linearly mixed statistically independent latent components in an unsupervised fashion. The ICA technique has advanced many tasks such as blind speech/audio separation and brain signal denoising. Since the late 1990s, attempts have been made towards generalizing the classic linear mixture model (LMM)-based ICA to nonlinear mixture models, e.g., in (Hyvarinen, 1999; Taleb & Jutten, 1999; Ziehe et al., 2003; Oja, 1997), driven by the ubiquity of nonlinearity in real-world data.

Formally, the *nonlinear independent component analysis* (nICA) problem deals with scenarios where statistically independent latent components are mixed by *unknown* nonlinear functions. The nICA technique aims at recovering the latent components up to certain (inconsequential) ambiguities. The nICA task finds many connections to modern machine learning and unsupervised representation learning problems. For example, a number of works used the nICA perspective to develop deep neural feature extractors for latent factor disentanglement (Bengio et al., 2013; Locatello et al., 2019; Higgins et al., 2017; Kim & Mnih, 2018; Chen et al., 2018). The disentanglement perspective was further connected to causal factor learning (Peters et al., 2017; Zhang & Hyvärinen, 2009; Monti et al., 2020). Furthermore, nICA and its close relatives were also used to understand popular neural representation learning frameworks such as contrastive learning (Hyvarinen & Morioka, 2016; 2017; Hyvarinen et al., 2019), variational autoencoder (VAE) (Khemakhem et al., 2020) and data-augmented self-supervised learning (Zimmermann et al., 2021). For all these tasks, nICA offers theory-driven perspectives to understand their successes and sometimes to improve their learning methods. In particular, the *identifiability* of the latent independent components under nICA models can often provide useful insights into these aspects.

The (n)ICA identifiability problem is concerned with the conditions and learning criteria under which one can reverse the unknown mixing process to recover the latent

components. However, unlike the classic LMM-based ICA whose model identifiability is well-studied (Comon, 1994; Hyvärinen & Oja, 2000; Comon & Jutten, 2010), identifiability of nICA models had not been fully understood for a long period. In fact, it is well-known that an nICA model that only assumes statistical independence of the latent components is not identifiable (Hyvärinen & Pajunen, 1999).

In recent years, a number of new nICA paradigms emerged, which nicely addressed the identifiability challenge under some additional yet physically meaningful conditions. These paradigms judiciously utilized structural information about the latent components, e.g., temporal dependence (Sprekeler et al., 2014; Hyvarinen & Morioka, 2017) and non-stationarity (Hyvarinen & Morioka, 2016) of data, to underpin the model identifiability. In particular, the work in (Hyvarinen et al., 2019) unified these developments under a *generalized contrastive learning* (GCL) based framework (Gutmann & Hyvärinen, 2010), where the latent components are assumed to be conditionally independent given an auxiliary variable. Under the GCL framework, the latent components are identifiable up to component-wise invertible nonlinear transformations, by simply learning a logistic regression neural discriminator.

The surprising connection between contrastive learning and nICA is both refreshing and insight-revealing. The identifiability proofs in (Hyvarinen & Morioka, 2016; 2017; Hyvarinen et al., 2019) are also elegant. Nevertheless, a caveat is that the GCL framework, same as other identifiable nICA works (Khemakhem et al., 2020; Locatello et al., 2020; Gresele et al., 2019), assumes that unlimited data samples are available. This presents a non-negligible gap between nICA identifiability theory and practice, since one never has unlimited data in real systems. In addition, the GCL-based nICA works all assume that universal exact function learners are used in their learning process. However, function approximation errors always exist in practice, even if very expressive function learners, e.g., deep/wide neural networks, are used. These less realistic assumptions naturally lead to an inquiry: Can the identifiability of GCL-based nICA be established under limited sample cases in the presence of learning function approximation errors?

Filling this theory-practice gap is a nontrivial task. First, the proofs in (Hyvarinen et al., 2019; Hyvarinen & Morioka, 2016; 2017) heavily rely on the equivalence between the optimal logistic regressor and the log-probability density function (log-PDF) difference of the two contrastive classes *in the limit of infinite samples*. Second, many steps in the proofs use first-order and second-order derivatives with respect to the learned latent components—whose existence over continuous open domains is also a result of the (uncountably) unlimited data assumption. In addition, unlike supervised learning where well-established generalization analysis routines (see, e.g., (Shalev-Shwartz & Ben-David, 2014)) can be used to characterize finite sample performance, there is no such toolkit for latent component analysis. This is perhaps because supervised learning's success is measured by the "distance" between an empirical loss and its population version, but latent component identification often has a much more intricate objective—which varies across different generative models and learning goals.

1.1. Contributions

In this work, our interest lies in offering a finite-sample analysis for the GCL-based nICA framework. Our analytical framework consists of three major steps. First, we use the notion of restricted strong convexity of the logistic loss used in GCL to characterize the relationship between its optimal solution and gradient at the optimum. Second, we combine statistical generalization theorems with this relation to characterize the gap between the regressor learned from finite samples and the optimum under the population case. Third, based on the gap, we characterize the separability of different latent components using numerical differentiation tools. As a result, we show that GCL-based nICA can separate different latent components to a reasonable extent under finite samples, and the performance improves when the sample size grows. Our result also takes the learning function's approximation error into consideration, and reveals an intuitive trade-off between the complexity and expressiveness of the employed learning function. To our best knowledge, the result is the first to establish such a finite sample identifiability under the GCL-based nICA framework. We also envision that our proof technique could help understand the finite-sample performance of different but nICA-related frameworks, e.g., those in (Zimmermann et al., 2021; Khemakhem et al., 2020; Locatello et al., 2020).

1.2. Notation

We use the following notations: $x, \boldsymbol{x}, \boldsymbol{X}$ represent a scalar, vector, and matrix, respectively; f' and f'' denote the first-order and second-order derivatives of function f, respectively; $f \circ g$ denotes the function composition of f and g; $\sigma_{\min}(\boldsymbol{W})$ denotes the smallest nonzero singular value of matrix $\boldsymbol{W}$; $\mathbb{E}[\cdot]$ denotes expectation of its argument; $p(x)$ denotes the probability density function of random variable x; a column vector $\boldsymbol{a} \in \mathbb{R}^D$ is defined as $\boldsymbol{a} = [a_1, \ldots, a_D]^\top = (a_1, \ldots, a_D)$.

We will frequently use the notation $\boldsymbol{x}_\ell \in \mathcal{X}$ to represent the ℓth sample drawn from a distribution $\mathcal{D}$ defined over the continuous domain $\mathcal{X}$. The notation $\boldsymbol{x}$ without subscript represents a random vector defined over the continuous domain $\mathcal{X}$ following the same distribution $\mathcal{D}$—i.e., $\boldsymbol{x}_\ell$ can be considered as the ℓth realization of the random vector $\boldsymbol{x}$.

2. Background

2.1. ICA, nICA, and Model Identifiability

The classic ICA techniques deal with the LMM, i.e.,

$$\boldsymbol{x}_\ell = \boldsymbol{A}\boldsymbol{s}_\ell,\ \ell = 1, \ldots, N, \tag{1}$$

where $\boldsymbol{x}_\ell \in \mathbb{R}^M$ is the ℓth observed sample, $\boldsymbol{s}_\ell \in \mathbb{R}^D$ are the D latent components, and $\boldsymbol{A} \in \mathbb{R}^{M\times D}$ with $M \geq D$. It is assumed that $\boldsymbol{x}_\ell$ and $\boldsymbol{s}_\ell$ are the ℓth realizations of the random vectors

$$\boldsymbol{x} = [x_1, \ldots, x_M]^\top, \text{ and } \boldsymbol{s} = [s_1, \ldots, s_D]^\top,$$

respectively, in which $\boldsymbol{x} = \boldsymbol{A}\boldsymbol{s}$ and $s_1, \ldots, s_D$ are statistically independent. The task of ICA is to recover $\boldsymbol{s}_\ell$ from $\boldsymbol{x}_\ell$. Recovering the latent components from LMMs is in general not possible, since $\boldsymbol{x}_\ell = \boldsymbol{A}\boldsymbol{s}_\ell = \boldsymbol{A}\boldsymbol{Q}\boldsymbol{Q}^{-1}\boldsymbol{s}_\ell$ for any nonsingular $\boldsymbol{Q}$. However, using the statistical mutual independence among $s_1, \ldots, s_D$, one can show that $\boldsymbol{s}_\ell$ and $\boldsymbol{A}$ are identifiable through ICA techniques up to permutation and scaling ambiguities. This is normally done by finding an inverse filter $\boldsymbol{W}$ such that the elements of $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}$ are mutually independent; see (Comon, 1994; Hyvärinen & Oja, 2000; Arora et al., 2012).

In nICA, the LMM in (1) is generalized to a nonlinear mixture model, i.e., (Hyvärinen & Pajunen, 1999)

$$\boldsymbol{x}_\ell = \boldsymbol{g}(\boldsymbol{s}_\ell), \tag{2}$$

where $\boldsymbol{g}(\cdot)$ is a smooth and invertible *unknown function*, and $\boldsymbol{x}_\ell$ and $\boldsymbol{s}_\ell$ are defined as before. The goal often amounts to learning a nonlinear function $\boldsymbol{h}(\cdot)$ such that for any $\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{s})$ the following holds:

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}) \text{ s.t. } y_i = \sigma_i(s_{\pi(i)}), \quad i = 1, \ldots, D, \tag{3}$$

where $\{\pi(1), \ldots, \pi(D)\}$ represents a permutation of $\{1, \ldots, D\}$ and $\sigma_i(\cdot) : \mathbb{R} \to \mathbb{R}$ is an unknown invertible function. Note that such y_i and $s_{\pi(i)}$ attain the maximum mutual information and can be converted from one to another. Thus, the learning goal is meaningful. Unfortunately, unlike ICA, under such a nonlinear mixture model, the desired y_i in (3) is in general not identifiable by just constraining the output of a learning system to be statistically independent (Hyvärinen & Pajunen, 1999)

2.2. Auxiliary Variable-Assisted GCL-based nICA

In recent years, some notable breakthroughs of the identifiability research of nICA have been made. Specifically, several recent works show that $\boldsymbol{s}$ in (2) can be identified (i.e., (3) can be guaranteed) under interesting and physically meaningful conditions; see, e.g., (Hyvarinen et al., 2019; Hyvarinen & Morioka, 2017; 2016). In particular, (Hyvarinen et al., 2019) distilled the essence and presented a unified framework based on GCL. Under the framework, it is assumed that $s_1, \ldots, s_D$ are statistically independent *conditioned on* the revelation of an *auxiliary variable* $\boldsymbol{u}$, i.e.,

$$\log p(\boldsymbol{s}|\boldsymbol{u}) = \sum_{i=1}^{D} q_i(s_i, \boldsymbol{u}), \tag{4}$$

where $q_i(\cdot)$ is a certain continuous function and $p(\boldsymbol{s}|\boldsymbol{u})$ is the conditional PDF of $\boldsymbol{s}$ given $\boldsymbol{u}$. Note that $\boldsymbol{u}$ is observed together with $\boldsymbol{x}$, i.e., $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{u})$ appears as a pair.

To put into context, we briefly mention some examples where the existence of auxiliary variables makes sense:

Example - Time Contrastive Learning (TCL) In TCL, the auxiliary variable could be an indicator of the time stamp with $u = t$ or other information, e.g., the mean and variance of of $\boldsymbol{s}_\ell$ in a certain time frame (Hyvarinen & Morioka, 2016; 2017; Hyvarinen et al., 2019). Here, the condition (4) means that the latent variables are independent if they are from the same time frame/slot.

Example - Multiview Contrastive Learning (MVCL) A slightly different but closely related example is MVL. There, $\boldsymbol{s}$ is assumed to be mixed with different nonlinear functions for each view (Gresele et al., 2019), i.e, $\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{s})$ with the auxiliary variable $\boldsymbol{u} \approx \widetilde{\boldsymbol{g}}(\boldsymbol{s})$ that is another view of the same data entity with some random perturbations. In this case, the log PDF $\log p(\boldsymbol{s}|\boldsymbol{u})$ can also be similarly factored as in (4) given that $s_1, \ldots, s_D$ are statistically independent.

Under (4), the nICA framework in (Hyvarinen et al., 2019) proposed to learn a regression function

$$r(\boldsymbol{x}, \boldsymbol{u}) = \sum_{i=1}^{D} \phi_i\left(h_i\left(\boldsymbol{x}\right), \boldsymbol{u}\right) \tag{5}$$

to distinguish between two types of $\boldsymbol{z}_\ell$, i.e.,

$$\boldsymbol{z}_\ell = (\boldsymbol{x}_\ell, \boldsymbol{u}_\ell) \text{ and } \boldsymbol{z}_\ell = (\boldsymbol{x}_\ell, \widetilde{\boldsymbol{u}}_\ell). \tag{6}$$

Note that the "positive samples" $\boldsymbol{z}_\ell = (\boldsymbol{x}_\ell, \boldsymbol{u}_\ell)$ are observed from data—and $\boldsymbol{u}_\ell$ is the natural auxiliary variable of $\boldsymbol{x}_\ell$. However, for the "negative samples" $\boldsymbol{z}_\ell = (\boldsymbol{x}_\ell, \widetilde{\boldsymbol{u}}_\ell)$, $\widetilde{\boldsymbol{u}}_\ell$ is randomly drawn from $p(\boldsymbol{u})$ which has no dependence on $\boldsymbol{x}_\ell$. For example, in TCL, the positive sample $\boldsymbol{z}_\ell = (\boldsymbol{x}_\ell, \boldsymbol{u}_\ell) = (\boldsymbol{x}_\ell, t)$ where $\ell \in$ time frame t, but the negative sample $\boldsymbol{z}_\ell = (\boldsymbol{x}_\ell, \widetilde{\boldsymbol{u}}_\ell) = (\boldsymbol{x}_\ell, t')$ where $\ell \notin$ time frame t'. In MVCL, the positive sample $\boldsymbol{z}_\ell = (\boldsymbol{x}_\ell, \boldsymbol{u}_\ell)$ which means that the two views are generated from the same $\boldsymbol{s}_\ell$. The negative sample $\boldsymbol{z}_\ell = (\boldsymbol{x}_\ell, \widetilde{\boldsymbol{u}}_\ell = \boldsymbol{u}_j)$ with $\ell \neq j$; i.e., the pair of data from the two views correspond to different latent vectors $\boldsymbol{s}_\ell$ and $\boldsymbol{s}_j$.

The functions h_i and ϕ_i are often represented by nonlinear function learners, e.g., neural networks. This "classification

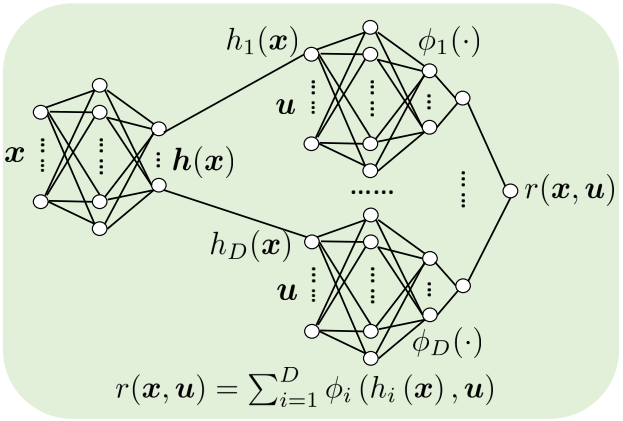

Figure 1: The neural network structure used for GCL-based nICA (Hyvarinen et al., 2019).

problem" can be realized using a logistic loss:

$$\min_{\boldsymbol{\phi},\boldsymbol{h}} \mathcal{L} = \min_{\boldsymbol{\phi},\boldsymbol{h}} \mathbb{E}_{\boldsymbol{z}} \left[\log(1+\exp[-dr(\boldsymbol{z})])\right], \tag{7}$$

where $d \in \{+1,-1\}$ is the "label" of $\boldsymbol{z}$. The realizations of d, namely, d_ℓ for $\ell = 1,\ldots,N$, are created using the following rule:

$$d_\ell = \begin{cases} +1, & \boldsymbol{z}_\ell = (\boldsymbol{x}_\ell, \boldsymbol{u}_\ell), \\ -1, & \boldsymbol{z}_\ell = (\boldsymbol{x}_\ell, \widetilde{\boldsymbol{u}}_\ell). \end{cases} \tag{8}$$

The learning system is shown in Fig. 1.

The framework is called "constrastive learning" because it constructs a discriminator from data itself without using traditional class labels as in supervised learning. Similar ideas are widely used in the domain of self-supervised representation learning (Chen et al., 2020; He et al., 2020; Tian et al., 2020; Oord et al., 2018).

The work in (Hyvarinen et al., 2019) showed that the learned $\boldsymbol{h}^\star(\boldsymbol{x})$ (i.e., the optimal solution of (27) in Appendix B) is the desired latent component $\boldsymbol{s}$ up to some ambiguities. To see the result, let us define the following

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}), \quad \boldsymbol{v}(\boldsymbol{y}) = \boldsymbol{g}^{-1}(\boldsymbol{h}^{-1}(\boldsymbol{y})) = \boldsymbol{s}.$$

Using the above notations, we restate the main result of (Hyvarinen et al., 2019) as follows:

Theorem 2.1 (Infinite-Sample Identifiability). *(Hyvarinen et al., 2019) Assume*

(i) that the data follows the model in (2) and (4) with $M = D$; the conditional log-PDF q_i in (4) is smooth (i.e., second-order differentiable) as a function of s_i for any $\boldsymbol{u}$;

*(ii) (**Variability Assumption**) that for any $\boldsymbol{y} \in \mathbb{R}^D$, there exist $2D+1$ vector $\boldsymbol{u}_j$'s, such that the $2D$ vectors in $\mathbb{R}^{2D}$ denoted as*

$$\boldsymbol{W} = [\boldsymbol{w}(\boldsymbol{y},\boldsymbol{u}_1) - \boldsymbol{w}(\boldsymbol{y},\boldsymbol{u}_0), \cdots, \boldsymbol{w}(\boldsymbol{y},\boldsymbol{u}_{2D}) - \boldsymbol{w}(\boldsymbol{y},\boldsymbol{u}_0)] \tag{9}$$

are linearly independent, where

$$\boldsymbol{w}(\boldsymbol{y},\boldsymbol{u}) = \left[\frac{\partial q_1(y_1,\boldsymbol{u})}{\partial y_1}, \cdots, \frac{\partial q_D(y_D,\boldsymbol{u})}{\partial y_D}, \frac{\partial^2 q_1(y_1,\boldsymbol{u})}{\partial y_1^2}, \cdots, \frac{\partial^2 q_D(y_D,\boldsymbol{u})}{\partial y_D^2}\right]; \tag{10}$$

(iii) that (7) is solved with universal function approximators to represent $r(\boldsymbol{z})$;

(iv) and that the learned optimal $\boldsymbol{h}^\star = (h_1^\star, \cdots, h_D^\star)$ is constrained to be invertible and smooth.

Then, in the limit of infinite data, we have $h^\star_{\pi(i)}(\boldsymbol{x}) = v_i^{-1}(s_i)$, for $i = 1,\ldots,D$, where $\{\pi(1),\ldots,\pi(D)\}$ is a permutation of $\{1,\ldots,D\}$.

The variability assumption means that the auxiliary variable $\boldsymbol{u}$ must provide sufficiently different information and impacts on the independent components to identify—i.e., the realizations of $\boldsymbol{u}$ should be diverse; see more explanations in (Hyvarinen et al., 2019; Gresele et al., 2019).

The proof of Theorem 2.1 consists of three major steps. **Step 1**: Under the unlimited data assumption, the optimal logistic regression function is converged to the log-density difference of the two classes (Goodfellow et al., 2014). To be specific, one can show that

$$r^\star(\boldsymbol{x},\boldsymbol{u}) = \sum_{i=1}^{D} \phi_i^\star\left(h_i^\star\left(\boldsymbol{x}\right),\boldsymbol{u}\right) = \log p(\boldsymbol{x}|\boldsymbol{u}) - \log p(\boldsymbol{x}), \tag{11}$$

if $r^\star$ is learned using a universal function approximator.

Step 2: Using the assumption in (4), a functional equation can be established everywhere over $\mathcal{X}$ (i.e., the domain of $\boldsymbol{x}$) by equating the constructed (5) and (11).

Step 3: Given the equation holds everywhere, cross-derivatives w.r.t. y_j and y_k are taken, which results in a linear system with a full-rank coefficient matrix under the assumption of Variability—which finally leads to the desired result.

We have restated the proof in Appendix C.1.1, as it may help the readers better understand the finite-sample analysis.

One can see that all the three steps heavily rely on the unlimited sample assumption. In the next section, we will offer an analytical framework that can circumvent this unrealistic assumption.

3. Finite-Sample Analysis of nICA

In practice, instead of directly solving (7), one always deals with the corresponding empirical loss function as follows:

$$\min_{\boldsymbol{\phi},\boldsymbol{h}} \widehat{\mathcal{L}} = \min_{\boldsymbol{\phi},\boldsymbol{h}} \frac{1}{N}\sum_{\ell=1}^{N} \log\left(1+\exp\left[-d_\ell r(\boldsymbol{z}_\ell)\right]\right). \quad (12)$$

Before stating the main results, we first define the vector that we hope to characterize as

$$\boldsymbol{\gamma}_{jk} = \left[\frac{\partial^2 v_1(\boldsymbol{y})}{\partial y_j \partial y_k}, \cdots, \frac{\partial^2 v_D(\boldsymbol{y})}{\partial y_j \partial y_k}\right]^\top. \quad (13)$$

The vector $\boldsymbol{\gamma}_{jk}$ will be used as a key metric to quantify the latent component identification performance. Fact 3.1 shows the rationale behind using such a vector to serve as our success metric—i.e., in the population case, $\boldsymbol{\gamma}_{jk} = \mathbf{0}$ holds for all (j,k) pairs where $j < k$ everywhere.

Fact 3.1. *In the proof of Theorem 2.1, it is noted that if $\boldsymbol{\gamma}_{jk} = \mathbf{0}$ for all (j,k)'s, then we have $h^\star_{\pi(i)}(\boldsymbol{x}) = v_i^{-1}(s_i)$, for $i = 1,\ldots,D$, where $\{\pi(1),\ldots,\pi(D)\}$ is a permutation of $\{1,\ldots,D\}$.*

Proof: First note that $\boldsymbol{\gamma}_{jk} = \mathbf{0}$ means that

$$\frac{\partial^2 v_i(\boldsymbol{y})}{\partial y_j \partial y_k} = 0 \quad (14)$$

for any $i \in \{1,\cdots,D\}$. Since we have $\boldsymbol{v}(\boldsymbol{y}) = \boldsymbol{g}^{-1}(\boldsymbol{h}^{-1}(\boldsymbol{y}))$ where both $\boldsymbol{g}$ and $\boldsymbol{h}$ are smooth and invertible functions, $\boldsymbol{v} = \boldsymbol{g}^{-1}\circ\boldsymbol{h}^{-1}$ is also invertible, which leads to the fact that the Jacobian $\boldsymbol{J}_{\boldsymbol{v}}$ should be full-rank:

$$\operatorname{rank}(\boldsymbol{J}_{\boldsymbol{v}}) = D. \quad (15)$$

Meanwhile, (14) implies that any v_i only depends on one of its arguments y_j. That is, the Jacobian must have the following form

$$\boldsymbol{J}_{\boldsymbol{v}} = \operatorname{Diag}(\boldsymbol{\lambda})\boldsymbol{\Pi}$$

where $\lambda_i \neq 0$ for $i = 1,\cdots,D$ and $\boldsymbol{\Pi}$ is a permutation matrix.

Note that it is impossible that different v_i and $v_{i'}$ are functions of the same y_j—otherwise the Jacobian would have at least a zero column, which violates $\operatorname{rank}(\boldsymbol{J}_v) = D$ in (15).

As a result, we have $h^\star_{\pi(i)}(\boldsymbol{x}) = v_i^{-1}(s_i)$ for $i = 1,\ldots,D$. ■

Fact 3.1 indicates that the "size" of $\|\boldsymbol{\gamma}_{jk}\|$ could quantify the level of success for separating the functions of $s_1,\ldots,s_D$. Hence, under the finite sample scenario, our goal is to show that $\|\boldsymbol{\gamma}_{jk}\|$ is bounded by $O((1/N)^\beta)$ for a certain $\beta > 0$.

To start with, we first characterize the Rademacher complexity of the neural network function class used to model the regression function $r(\boldsymbol{z})$. We define the function class $\mathcal{F}$ for multi-layer perceptrons (MLP).

Assumption 3.2 (Neural Network)**.** Assume that $\boldsymbol{h}(\cdot) : \mathbb{R}^D \to \mathbb{R}^D$ and each $\phi_i(\cdot) : \mathbb{R}^D \to \mathbb{R}$ is parameterized by an L-layer neural network with the following structure and constraint

$$\mathcal{F} = \{\boldsymbol{f} | \boldsymbol{f}(\boldsymbol{z}) = \boldsymbol{P}_L\boldsymbol{\zeta}(\ldots\boldsymbol{P}_2\boldsymbol{\zeta}(\boldsymbol{P}_1\boldsymbol{z})),\ \|\boldsymbol{P}_i\|_F \leq B_i\}, \quad (16)$$

where the activation function $\boldsymbol{\zeta}(\cdot) = [\zeta_1(\cdot),\ldots,\zeta_{D_j}(\cdot)]^\top$, and $\zeta_i(\cdot) : \mathbb{R} \to \mathbb{R}$ is a 1-Lipschitz continuous function that satisfies $\zeta_i(0) = 0$ for $i = 1,\ldots,D_j$ for $j = 1,\ldots,L$, and D_j is the network width of the jth layer.

We hope to remark that the identifiability theory in (Hyvarinen et al., 2019) and this work require that $\boldsymbol{h}$ to be invertible. Such $\boldsymbol{h}$ could be approximated using special networks such as normalizing flows or autoencoder-type regularization. Nonetheless, we use a generic neural network following (Hyvarinen et al., 2019), which suggested that such simple constructions of $\boldsymbol{h}$ normally do not hurt the performance.

We have the following bound for the complexity of $\mathcal{F}$.

Lemma 3.3 ((Golowich et al., 2018), Corollary 1)**.** *Assume that the observation is bounded as $\|\boldsymbol{z}\|_2 \leq C$. The Rademacher complexity $\mathfrak{R}_N^f$ of the function class defined in 3.2 is bounded by*

$$\mathfrak{R}_N^f \leq O\left(C\prod_{i=1}^{L} B_i \min\left\{\frac{\log^{3/4}(N)\sqrt{\log\left(\frac{C}{\Gamma}\prod_{i=1}^{L} B_i\right)}}{N^{1/4}}, \sqrt{\frac{L}{N}}\right\}\right), \quad (17)$$

where $\sup_{\boldsymbol{z}\in\mathcal{Z}} |r(\boldsymbol{z})| \geq \Gamma$.

The bound can be simplified to $\mathfrak{R}_N^f \leq O(C\prod_{i=1}^{L} B_i\sqrt{L/N})$.

Under this simplification, we have the following complexity bound for $r(\boldsymbol{z})$:

Lemma 3.4. *Assume that the observation is bounded as $\|\boldsymbol{x}\|_2 \leq C_x$ and $\|\boldsymbol{u}\|_2 \leq C_u$. Then the Rademacher complexity $\mathfrak{R}_N$ of $r(\boldsymbol{z})$ is bounded by*

$$\mathfrak{R}_N \leq O\left(\left[C_x\prod_{i=1}^{L} B_i + \sqrt{D}C_u\right]\prod_{i=1}^{L} B_i\sqrt{\frac{DL}{N}}\right).$$

The detailed proof is in Appendix A.

Under the population case, Step 1 in the proof of Theorem 2.1 assumed that $r^\star(\boldsymbol{z})$ equals to the log of the PDF differences of the positive and negative classes [cf. Appendix C.1.1]. Then, the next steps can take derivatives using this equation over the continuous domain where $\boldsymbol{z}$ is defined. However, with N samples, the distance between the learned regression function and the log-PDF difference is not zero. To characterize this distance, we introduce the following lemma:

Lemma 3.5. *Assume that $|r(\boldsymbol{z})| \leq \alpha$ over all $\boldsymbol{z} \in \mathcal{Z}$. The logistic function $\ell(r) = \log(1 + e^{-dr})$ is γ_α-restricted strongly convex (γ_α-RSC) in r, where $\gamma_\alpha = \frac{e^\alpha}{(1+e^\alpha)^2}$.*

Proof: Starting with the following function for any $(\boldsymbol{x}, \boldsymbol{u})$

$$\ell(r) = \log(1 + e^{-dr}),$$

we take derivative w.r.t. r, which gives us

$$\frac{d\ell(r)}{dr} = \frac{-d}{1 + e^{dr}}, \quad \frac{d^2\ell(r)}{dr^2} = \frac{e^{dr}}{(1 + e^{dr})^2}$$

since $d = -1$ or $d = 1$.

The function $\frac{e^{dr}}{(1+e^{dr})^2}$ is maximized when $r = 0$ and it is monotonic for $r < 0$ and $r > 0$. By assuming that $|r| \leq \alpha$, we have

$$\begin{aligned}\frac{e^{dr}}{(1 + e^{dr})^2} &\geq \min\left\{\frac{e^{-\alpha}}{(1 + e^{-\alpha})^2}, \frac{e^{\alpha}}{(1 + e^{\alpha})^2}\right\} \\ &= \frac{e^{\alpha}}{(1 + e^{\alpha})^2} = \frac{e^{-\alpha}}{(1 + e^{-\alpha})^2}\end{aligned}$$

Therefore, for bounded r, the logistic function is γ_α-strongly convex, with

$$\gamma_\alpha = \frac{e^{\alpha}}{(1 + e^{\alpha})^2},$$

which completes the proof. ∎

Note that the restricted strong convexity of the logistic loss was often mentioned in the one-bit matrix/tensor recovery literature; see, e.g., (Ni & Gu, 2016).

In (Hyvarinen et al., 2019), it was assumed that $\widehat{r}^\star(\boldsymbol{z})$ is a learning function that is an universal function approximator. Hence, $\widehat{r}^\star(\boldsymbol{z})$ can exactly express the desired function [i.e., a log-PDF difference as on the right hand side of (11)]. In practice, we take r from a function class $\mathcal{R}$ that may not be powerful enough to express any function, even if $\mathcal{R}$ is a deep neural network class. To take this function mismatch into consideration, we make the following assumption:

Assumption 3.6. Assume that we use $r \in \mathcal{R}$ as the learning function. The best learned $r \in \mathcal{R}$ from the population case (7) is characterized as

$$\min_{r \in \mathcal{R}} \mathcal{L}(r) - \mathcal{L}(r^\star) \leq \nu \tag{18}$$

where $r^\star$ is the desired regression function as in (11).

Note that the bound ν serves as an indicator of the expressiveness of the function class $\mathcal{R}$. For example, when $\mathcal{R}$ consists of neural networks, ν decreases as the neural network becomes deeper and wider.

Using Lemma 3.5, we show the following key lemma:

Lemma 3.7. *Assume that the empirical loss in (12) is trained with i.i.d. samples $\{\boldsymbol{z}_\ell\}_{\ell=1}^N$, and that the criterion is optimally solved. Also assume that the solution of (12) is taken from the function class $\mathcal{R}$. Then, we have the following bound over the domain $\boldsymbol{z} \in \mathcal{Z} = \mathcal{X} \times \mathcal{U}$ with probability at least $1 - \delta$:*

$$\begin{aligned}&\mathbb{E}_{\mathcal{D}}[|\widehat{r}^\star(\boldsymbol{z}) - r^\star(\boldsymbol{z})|^2] \\ &\leq \frac{(1 + e^\alpha)^2}{e^\alpha}\left(2\mathfrak{R}_N + \nu + 5c\sqrt{\frac{2\ln(8/\delta)}{N}}\right),\end{aligned} \tag{19}$$

where $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{u} \in \mathcal{U}$ in which $\mathcal{X}$ and $\mathcal{U}$ are two continuous open domains, $\mathcal{D}$ is the distribution where any $\boldsymbol{z}_\ell \in \mathcal{Z}$ is drawn from, $\widehat{r}^\star$ and $r^\star$ are the optimal solutions of (12) and the desired regression function in (11), respectively, $\alpha = \left(\sqrt{D}C_x \prod_{i=1}^L B_i + DC_u\right)\prod_{i=1}^L B_i$ is an upper bound of $|r(\boldsymbol{z})|$.

The proof of Lemma 3.7 can be found in Appendix B.

The bound in Lemma 3.7 shows the expected distance between $\widehat{r}^\star$ and $r^\star$—and how the gap scales with various problem parameters. This will be used in the next steps to estimate the numerical derivative in the proof of the main theorem:

Theorem 3.8 (Main Result). *Under the generative model (2) and Assumption 3.2, assume that the learning problem defined in Theorem 2.1 is solved with N i.i.d. samples $\{\boldsymbol{z}_\ell\}_{\ell=1}^N$, and that the learned $\boldsymbol{h}$ is invertible. Suppose that the learned $\widehat{r}^\star \in \mathcal{R}$ is fourth-order differentiable w.r.t. $\boldsymbol{y}$ and the absolute value of the fourth-order partial derivative of $t(\boldsymbol{z}) = \widehat{r}^\star(\boldsymbol{z}) - r^\star(\boldsymbol{z})$ w.r.t. any y_i is upper bounded by C_t. Then, we have the following bound with probability of at least $1 - \delta$,*

$$\begin{aligned}&\mathbb{E}_{\mathcal{D}}\left[\left\|\widehat{\boldsymbol{\gamma}}_{jk}\right\|_2^2\right] \\ &\leq O\left(\frac{DC_t(1 + e^\alpha)}{e^{\alpha/2}\sigma_*^2}\left(\mathfrak{R}_N + \nu + \alpha\sqrt{\frac{\ln(1/\delta)}{N}}\right)^{1/2}\right),\end{aligned} \tag{20}$$

where $\mathcal{D}$ is the distribution of $\boldsymbol{z}$, $\widehat{\boldsymbol{\gamma}}_{jk}$ is an estimation of $\boldsymbol{\gamma}_{jk}$ at any observed $\boldsymbol{x}_\ell$ using N samples for any (j, k) pair with $j < k$ where the upper bound $\alpha = (\sqrt{D}C_x \prod_{i=1}^L B_i + DC_u)\prod_{i=1}^L B_i$, and $\sigma_* = \max_{\boldsymbol{W}} \sigma_{\min}(\boldsymbol{W})$ [cf. the definition of $\boldsymbol{W}$ in (9)].

Theorem 3.8 asserts that with a large enough N, the $\widehat{\boldsymbol{\gamma}}_{jk}$'s are almost zero vectors for all (j, k)'s. As mentioned in Fact 3.1, this serves as a quantified indicator for the latent component identification performance.

The result in Theorem 3.8 is intuitive—it presents a trade-off between the learning function's complexity and its expressiveness. Specifically, given a certain N, increasing the

network complexity (e.g., by increasing the network depth and width) makes the learning function class $\mathcal{R}$ more expressive (i.e., with a smaller ν) but more complex (i.e., with a larger $\mathfrak{R}_N$). If N is not large, $\mathfrak{R}_N$ could dominate the right hand side of (20). Hence, under a small or moderate sample size, it is useful to employ a reasonably expressive network to serve as $\widehat{r}^\star$, but not encouraged to use an *overly* deep/wide neural network. This is similar to the widely recognized "data-hungry" and overfitting phenomena observed in supervised *deep* learning problems.

Proof Sketch. We sketch the proof of Theorem 3.8 here—the readers are referred to the appendices for the detailed proof. Our proof consists of the following steps.

First, starting from (12), we estimate the performance of the learned regression function; i.e., we bound the following distance

$$\mathbb{E}_{\mathcal{D}}[|\widehat{r}^\star(\boldsymbol{z}) - r^\star(\boldsymbol{z})|^2],$$

which can be done by invoking Lemma 3.5 and Lemma 3.7.

Second, we construct

$$t(\boldsymbol{y}_\ell) = \sum_{i=1}^{D} q_i(v_i(\boldsymbol{y}_\ell), \boldsymbol{u}_\ell) - \log p_s(\boldsymbol{v}(\boldsymbol{y}_\ell)) - \sum_{i=1}^{D} \phi_i\left([\boldsymbol{y}_\ell]_i, \boldsymbol{u}_\ell\right). \tag{21}$$

at any observed sample ℓ—which is a sample version of the key functional equation for establishing the infinite sample nICA identifiability; see more details in Appendix C.1.1.

Taking numerical derivatives of (21) w.r.t. y_j and y_k, we establish a "noisy" system of linear equation

$$\boldsymbol{W}\boldsymbol{\kappa}_{jk} = \boldsymbol{b} \approx \boldsymbol{0},$$

where $\|\boldsymbol{b}\|$ can be characterized by the quantity of $\mathbb{E}[|\widehat{r}^\star(\boldsymbol{z}) - r^\star(\boldsymbol{z})|^2]$ and $\boldsymbol{\kappa}_{jk}$ includes $\widehat{\gamma}_{jk}$ as a sub-vector.

Third, combining with the smallest singular value of $\boldsymbol{W}$, using standard perturbation analysis of system of linear equations can help estimate the upper bound of $\widehat{\gamma}_{jk}$. ■

The full proof is relegated to Appendix C.

4. Related Works

The work in (Arora et al., 2012) considered finite sample analysis for the classic ICA under the LMM. The recent work in (Lyu & Fu, 2021) considered finite sample analysis of post-nonlinear mixture model. However, the post-nonlinear mixture model is a special kind of simplified nonlinear model, whose finite-sample analysis is much less challenging than the case in this work. In (Lyu et al., 2022; Lyu & Fu, 2022), sample complexity of latent component recovery was studied under the deep canonical correlation analysis and self-supervised learning settings. The learning criteria there are different from GLS, and are arguably easier to handle due to their least squares nature. In contrastive learning, (Arora et al., 2019; Tosh et al., 2021; Tsai et al., 2020; HaoChen et al., 2021) analyzed finite sample performance, but in terms of downstream classification error. This is more aligned with traditional generalization error analysis, instead of latent component identification as in this work.

We should mention that our analysis is built upon the nICA framework in Theorem 1 of (Hyvarinen et al., 2019). There are also other nICA works in the literature. For example, (Zimmermann et al., 2021) showed that data-augmented contrastive learning can recover the latent components up to affine transformations, under some more specific conditions, e.g., the latent components' inner product follows a certain distribution. An exponential family distribution assumption on $\boldsymbol{s}$ was used in Theorem 3 of (Hyvarinen et al., 2019), which also helped connect nICA and VAE; see (Khemakhem et al., 2020). Our proof does not use assumptions on the distribution of $\boldsymbol{s}$.

5. Numerical Validation

In this section, we validate our theoretical results using synthetic and real data experiments.

5.1. Synthetic Data -TCL

Data Generation We follow the setup of the first experiment in (Hyvarinen et al., 2019) for TCL and consider time-domain data $\{\boldsymbol{x}_\ell\}_{\ell=1}^N$. We generate latent components $\boldsymbol{s} = [s_1, s_2]^\top \in \mathbb{R}^2$ that are divided into 5 different time frames. The latent component samples $[\boldsymbol{s}_\ell]_i$ for $i = 1, 2$ are generated using a distribution specified by $\boldsymbol{u}_\ell \in \{\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_5\}$; i.e., $\boldsymbol{u}_\ell$ are randomly drawn from 5 different vectors, each corresponding to a time frame. Specifically, $[\boldsymbol{s}_\ell]_i$ for $i = 1, 2$ and $\ell \in$ time frame τ are generated by the multiplication of a Gaussian distribution and a Laplacian distribution, and the mean and variance/scale information of the distributions are contained in $\boldsymbol{\omega}_\tau$. The multiplication is needed to meet the variability condition in Theorem 2.1; see more details in (Hyvarinen et al., 2019). The generative function $\boldsymbol{g}(\cdot) : \mathbb{R}^D \to \mathbb{R}^D$ is a one-hidden-layer neural network with leaky ReLU activations. The network weights are drawn from the standard Gaussian distribution. The vectors $\boldsymbol{z}_\ell = (\boldsymbol{x}_\ell, \boldsymbol{u}_\ell)$ for $\ell = 1, \ldots, N$ can be considered as i.i.d. samples that are re-arranged into different time frames. We run the TCL framework using different sample sizes N with equally divided time frames.

Evaluation Metrics The goal of nICA is to output $y_{\pi(i)} = v_i^{-1}(s_i)$ for $i = 1, \cdots, D$. Hence, to evaluate the performance, we use the *mutual information* (MI) between the

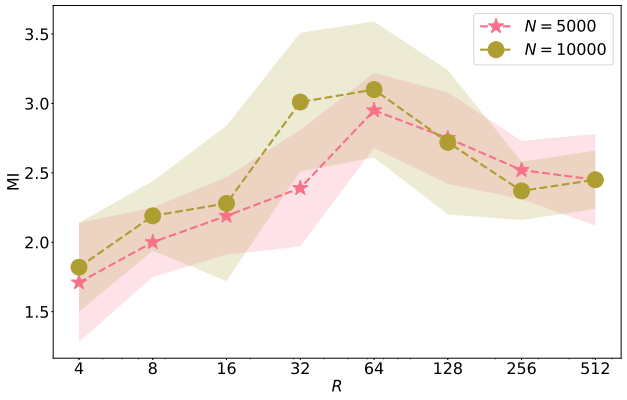

Figure 2: The MI performance under the TCL setting.

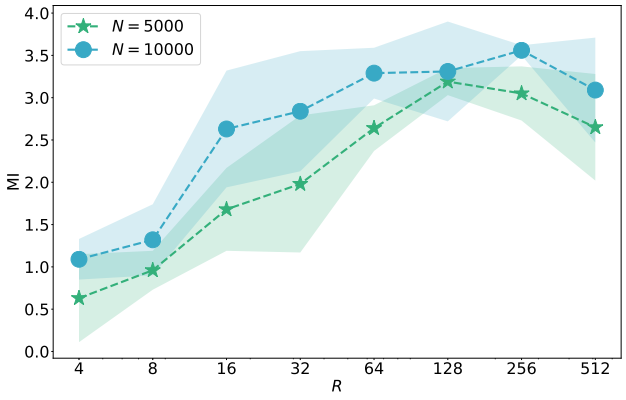

Figure 3: The MI performance under the MVCL setting.

estimated $\widehat{y}_{\pi(i)}$ and the corresponding s_i as our evaluation metric, since the MI of $\widehat{y}_{\pi(i)}$ and the associated s_i is maximized when $\widehat{y}_{\pi(i)} = v_i^{-1}(s_i)$ with an invertible $v_i(\cdot)$. Specifically, we estimate the MI using kernel density estimation (Kozachenko & Leonenko, 1987). We compute the MI between each of the recovered $\widehat{y}_i$ and the ground truth s_j's. Then, we use the Hungarian algorithm (Kuhn, 1955) to fix the permutation ambiguity.

Neural Network Settings We model $\boldsymbol{h}(\cdot)$ and $\phi_i(\cdot)$ using three-hidden-layer neural networks. We test various R's (i.e., the number of hidden neurons) for each layer, where $R \in \{4, 8, 16, 32, 64, 128, 256, 512\}$. The activation function is ReLU. Note that a larger R means a more complex (wider) neural network, which has a higher expressive power (Hornik, 1991; Hassoun et al., 1995) (i.e., a smaller ν) but leads to a larger Rademacher complexity $\mathfrak{R}_N$. For optimization, we use the `Adam` optimizer (Kingma & Ba, 2014) with an initial learning rate 5×10^{-4}.

Results Fig. 2 shows the nICA performance in terms of MI using different network width R's under $N = 5,000$ and $N = 10,000$. The results are averaged over 5 random trials. One can see that, under a given N, the MI performance improves when the network size R increases from 4 to 64. When R continues to grow, the MI performance shows a descending trend. This exactly reflects the expressiveness (ν) and complexity ($\mathfrak{R}_N$) trade-off revealed in Theorem 3.8. That is, the initial performance improvement is likely due to the fact that wider neural networks can better approximate the desired unknown functions, and the decrease of MI may be due to the fact that the overly complex neural networks have a dominating $\mathfrak{R}_N$.

5.2. Synthetic Data - MVCL

Data Generation In this subsection, we use the multiview setting from (Gresele et al., 2019)—also see the second example in Sec. 2.2. We use $\boldsymbol{x}_\ell = \boldsymbol{g}(\boldsymbol{s}_\ell)$ as the first view. We set $\boldsymbol{u}_\ell = \widetilde{\boldsymbol{g}}(s_\ell)$ to be data from another view. We use $D = 2$ with each component sampled from independent uniform distribution $\mathrm{U}[-a, a]$ with different a's. For $\boldsymbol{x}$, the mixing function $\boldsymbol{g}$ is a one-hidden-layer neural network, with D hidden neurons. For $\boldsymbol{u}_\ell$, the generation follows (Gresele et al., 2019) where $\boldsymbol{u}_\ell = \widetilde{\boldsymbol{g}}(\boldsymbol{s}_\ell + \boldsymbol{n}_\ell)$, in which $\boldsymbol{n}_\ell$ is again a product of a Gaussian variable and a Laplacian variable (Hyvarinen et al., 2019). Under this setting, the *sufficiently distinct views* condition in (Gresele et al., 2019) (which is derived from the variability assumption in Theorem 2.1) is satisfied. Similarly, $\widetilde{\boldsymbol{g}}$ is another one-hidden-layer neural network, with D hidden neurons. For both of $\boldsymbol{g}$ and $\widetilde{\boldsymbol{g}}$, the neural network coefficients are generated from standard normal distribution. For invertibility consideration, the activation function used is leaky ReLU. The positive and negative samples are generated by (6).

Metric and Neural Network Settings We continue using MI as our evaluation metric. The settings of $\boldsymbol{h}(\cdot)$ and $\phi_i(\cdot)$ are the same as those in the previous experiment.

Results Fig. 3 shows latent component identification performance evaluated by MI on the first view. The results are averaged over 5 random trials. Similar trends can be observed as in Fig. 2. That is, the performance improves when R increases from 4, but starts to decline for $R \geq 128$ when $N = 5000$ and $R \geq 256$ when $N = 10000$.

5.3. Real Data

Data and Settings In addition to synthetic data, we also use the "EEG eye" dataset from the UCI repository (Dua & Graff, 2017). The task here is to predict whether the eyes of the subject are open or closed based on the EEG recording $\boldsymbol{x}_\ell$ at time ℓ. The EEG data $\boldsymbol{x}_\ell$ can be considered as a nonlinear mixture of some latent signals $\boldsymbol{s}_\ell$ "emitted" by the brain. Hence, if one could learn the unmixed latent

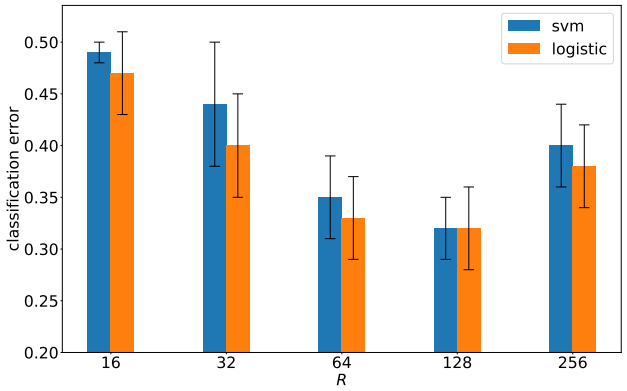

Figure 4: Test error on the EEG data under various R's.

components $\widehat{\boldsymbol{s}}_\ell = \boldsymbol{h}(\boldsymbol{x}_\ell)$ and use them as the extracted features of $\boldsymbol{x}_\ell$, it may help reduce the complexity of the learner in the downstream tasks (e.g., classification).

The data $\boldsymbol{x}_\ell$'s have fourteen dimensions. We aim to learn a five-dimensional underlying latent $\boldsymbol{s}_\ell$ for each data sample. We use 12,000 data samples as the training set to learn $\boldsymbol{h}(\cdot)$. Then, we train simple classifiers (i.e., SVM and logistic regression) using $\widehat{\boldsymbol{s}}_\ell = \boldsymbol{h}(\boldsymbol{x}_\ell)$. The classifiers are tested using 3000 test samples. We run the TCL framework in (Hyvarinen & Morioka, 2016) on the EEG data. We split the training data into 60 time frames with 200 samples within each frame. We use the frame label to be $\boldsymbol{u}_\ell$, where $\boldsymbol{u}_\ell \in \{0, \cdots, 59\}$. The vector $\boldsymbol{z}_\ell$ is constructed following the description in Sec. 2.2; also see (Gresele et al., 2019). The network structures of both $\boldsymbol{h}(\cdot)$ and $\phi_i(\cdot)$ are as before.

Results Fig. 4 shows the averaged classification errors using SVM and logistic regression, respectively. The results are averaged over 5 random trials. One can observe a similar phenomenon as seen in the synthetic experiments. In particular, the classification performance improves as R increases, because the function $\boldsymbol{h}(\cdot)$ becomes more expressive. But when the neural network gets overly complex (i.e., when $R > 128$), the performance deteriorates. This result again corroborates our main result in Theorem 3.8.

6. Conclusion

In this work, we investigated the identifiability problem of GCL-based nICA under a practical finite sample setting. The GCL-based nICA framework is an important development of the long-existing nonlinear latent component identification problem, yet existing works all used an infinite sample assumption to establish model identifiability. Our work is the first to address the identifiability problem under a finite sample setting, to our best knowledge. The proposed analytical framework is a nontrivial integration of properties of the logistic loss, the classic generalization theory of supervised learning, and numerical differentiation. Unlike the existing GCL-based nICA works that all assume the use of universal exact function learners to establish identifiability, our analysis also provides insights into the trade-off between the expressiveness and the complexity of the employed function approximators. We envision that the analytical framework can be further applied to a wider range of unsupervised/self-supervised learning problems for studying finite-sample identifiability.

Acknowledgement. This work is supported in part by the National Science Foundation (NSF) CAREER Award ECCS-2144889, and in part by the Army Research Office (ARO) under Project ARO W911NF-21-1-0227. We also thank the anonymous reviewers for their valuable comments and constructive suggestions.

References

Comon, P. and Jutten, C. *Handbook of Blind Source Separation*. Elsevier, 2010.

Arora, S., Ge, R., Moitra, A., and Sachdeva, S. Provable ICA with unknown Gaussian noise, with implications for Gaussian mixtures and autoencoders. In *Advances in Neural Information Processing Systems*, volume 25, 2012.

Arora, S., Khandeparkar, H., Khodak, M., Plevrakis, O., and Saunshi, N. A theoretical analysis of contrastive unsupervised representation learning. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

Bartlett, P. L. and Mendelson, S. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

Bengio, Y., Courville, A., and Vincent, P. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35 (8):1798–1828, 2013.

Chen, R. T., Li, X., Grosse, R. B., and Duvenaud, D. K. Isolating sources of disentanglement in variational autoencoders. In *Advances in Neural Information Processing Systems*, pp. 2610–2620, 2018.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, pp. 1597–1607. PMLR, 2020.

Comon, P. Independent component analysis, a new concept? *Signal Processing*, 36(3):287 – 314, 1994.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Golowich, N., Rakhlin, A., and Shamir, O. Size-independent sample complexity of neural networks. In *Conference On Learning Theory*, pp. 297–299. PMLR, 2018.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.

Gresele, L., Rubenstein, P. K., Mehrjou, A., Locatello, F., and Schölkopf, B. The incomplete Rosetta stone problem: Identifiability results for multi-view nonlinear ICA. In *Uncertainty in Artificial Intelligence*, pp. 217–227, 2019.

Gutmann, M. and Hyvärinen, A. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, pp. 297–304. JMLR Workshop and Conference Proceedings, 2010.

HaoChen, J. Z., Wei, C., Gaidon, A., and Ma, T. Provable guarantees for self-supervised deep learning with spectral contrastive loss. In *Advances in Neural Information Processing Systems*, 2021.

Hassoun, M. H. et al. *Fundamentals of artificial neural networks*. MIT press, 1995.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738, 2020.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. Beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017.

Hornik, K. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991.

Hyvarinen, A. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks,*, 10(3):626–634, 1999.

Hyvarinen, A. and Morioka, H. Unsupervised feature extraction by time-contrastive learning and nonlinear ICA. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Hyvarinen, A. and Morioka, H. Nonlinear ICA of temporally dependent stationary sources. In *International Conference on Artificial Intelligence and Statistics*, volume 54, pp. 460–469, 20–22 Apr 2017.

Hyvärinen, A. and Oja, E. Independent component analysis: Algorithms and applications. *Neural Networks*, 13(4-5): 411–430, 2000.

Hyvärinen, A. and Pajunen, P. Nonlinear Independent Component Analysis: Existence and uniqueness results. *Neural Networks*, 12(3):429–439, 1999.

Hyvarinen, A., Sasaki, H., and Turner, R. Nonlinear ICA using auxiliary variables and generalized contrastive learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 859–868, 2019.

Khemakhem, I., Kingma, D., Monti, R., and Hyvarinen, A. Variational autoencoders and nonlinear ICA: A unifying framework. In *International Conference on Artificial Intelligence and Statistics*, pp. 2207–2217. PMLR, 2020.

Kim, H. and Mnih, A. Disentangling by factorising. In *International Conference on Machine Learning*, volume 80, pp. 2649–2658. PMLR, 10–15 Jul 2018.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kozachenko, L. and Leonenko, N. N. Sample estimate of the entropy of a random vector. *Problemy Peredachi Informatsii*, 23(2):9–16, 1987.

Kuhn, H. W. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2): 83–97, 1955.

Locatello, F., Bauer, S., Lucic, M., Raetsch, G., Gelly, S., Schölkopf, B., and Bachem, O. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning*, pp. 4114–4124. PMLR, 2019.

Locatello, F., Poole, B., Rätsch, G., Schölkopf, B., Bachem, O., and Tschannen, M. Weakly-supervised disentanglement without compromises. In *International Conference on Machine Learning*, pp. 6348–6359. PMLR, 2020.

Lyu, Q. and Fu, X. Identifiability-guaranteed simplex-structured post-nonlinear mixture learning via autoencoder. *IEEE Transactions on Signal Processing*, 69:4921–4936, 2021.

Lyu, Q. and Fu, X. Finite-sample analysis of deep cca-based unsupervised post-nonlinear multimodal learning. *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–7, 2022.

Lyu, Q., Fu, X., Wang, W., and Lu, S. Understanding latent correlation-based multiview learning and self-supervision: An identifiability perspective. In *International Conference on Learning Representations*, 2022.

Monti, R. P., Zhang, K., and Hyvärinen, A. Causal discovery with general non-linear relationships using non-linear ICA. In *Uncertainty in Artificial Intelligence*, pp. 186–195. PMLR, 2020.

Mørken, K. Numerical algorithms and digital representation. *Lecture Notes for course MATINF1100 Modelling and Computations,(University of Oslo, Ch. 11, 2010)*, 2013.

Ni, R. and Gu, Q. Optimal statistical and computational rates for one bit matrix completion. In *Artificial Intelligence and Statistics*, pp. 426–434. PMLR, 2016.

Oja, E. The nonlinear PCA learning rule in Independent Component Analysis. *Neurocomputing*, 17(1):25–45, 1997.

Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Peters, J., Janzing, D., and Schölkopf, B. *Elements of causal inference: foundations and learning algorithms*. The MIT Press, 2017.

Shalev-Shwartz, S. and Ben-David, S. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Sprekeler, H., Zito, T., and Wiskott, L. An extension of slow feature analysis for nonlinear blind source separation. *The Journal of Machine Learning Research*, 15(1):921–947, 2014.

Taleb, A. and Jutten, C. Source separation in post-nonlinear mixtures. *IEEE Transactions on Signal Processing*, 47 (10):2807–2820, 1999.

Tian, Y., Krishnan, D., and Isola, P. Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pp. 776–794. Springer, 2020.

Tosh, C., Krishnamurthy, A., and Hsu, D. Contrastive learning, multi-view redundancy, and linear models. In *Algorithmic Learning Theory*, pp. 1179–1206. PMLR, 2021.

Tsai, Y.-H. H., Wu, Y., Salakhutdinov, R., and Morency, L.-P. Self-supervised learning from a multi-view perspective. In *International Conference on Learning Representations*, 2020.

Zhang, K. and Hyvärinen, A. On the identifiability of the post-nonlinear causal model. In *25th Conference on Uncertainty in Artificial Intelligence*, pp. 647–655, 2009.

Ziehe, A., Kawanabe, M., Harmeling, S., and Müller, K.-R. Blind separation of post-nonlinear mixtures using linearizing transformations and temporal decorrelation. *Journal of Machine Learning Research*, 4:1319–1338, 2003.

Zimmermann, R. S., Sharma, Y., Schneider, S., Bethge, M., and Brendel, W. Contrastive learning inverts the data generating process. In *Proceedings of the 38th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 18–24 Jul 2021.

A. Proof of Lemma 3.4

We show the Rademacher complexity of the network plotted in Fig. 1. Note that norm the output of $\|\boldsymbol{h}(\boldsymbol{x})\|_2$ is bounded by

$$\|\boldsymbol{h}(\boldsymbol{x})\|_2 \leq C_x \prod_{i=1}^{L} B_i \tag{22}$$

by simply using the Cauchy–Schwarz inequality. Next, assume that

$$|h_i(\boldsymbol{x})| \leq \frac{C_x \prod_{i=1}^{L} B_i}{\sqrt{D}}. \tag{23}$$

Then, for each of the ϕ_i network, the norm of its input is bound by

$$\sqrt{\frac{(C_x \prod_{i=1}^{L} B_i)^2}{D} + C_u^2}. \tag{24}$$

The Racemecher complexity of ϕ_i is bounded as

$$\sqrt{\frac{(C_x \prod_{i=1}^{L} B_i)^2}{D} + C_u^2} \prod_{i=1}^{L} B_i \sqrt{\frac{L}{N}}. \tag{25}$$

The final complexity of $r(\cdot)$ is the summation of the above, which is

$$\begin{aligned} \mathfrak{R}_N &\leq D \sqrt{\frac{(C_x \prod_{i=1}^{L} B_i)^2}{D} + C_u^2} \prod_{i=1}^{L} B_i \sqrt{\frac{L}{N}} \\ &\leq \left(C_x \prod_{i=1}^{L} B_i + \sqrt{D} C_u \right) \prod_{i=1}^{L} B_i \sqrt{\frac{DL}{N}} \end{aligned} \tag{26}$$

where the second inequality is by $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$ for $a > 0, b > 0$. ∎

B. Proof of Lemma 3.7

B.1. Bound the Generalization Error of the Regression Function

Define the following function:

$$\ell(\boldsymbol{z}, d; r) = \log(1 + \exp[-d r(\boldsymbol{z})])$$

Using the above notation, consider the following loss function:

$$\min_r \ \mathcal{L}(r) = \min_r \ \mathbb{E}\left[\ell(\boldsymbol{z}, d; r)\right] = \min_r \ \mathbb{E}\left[\log(1 + \exp[-d r(\boldsymbol{z})])\right] \tag{27}$$

where r is the nonlinear function to learn. The finite-sample version is as follows:

$$\min_r \ \widehat{\mathcal{L}}(r) = \min_r \ \frac{1}{N} \sum_{\ell=1}^{N} \ell(\boldsymbol{z}_\ell, d_\ell; r) = \min_r \ \frac{1}{N} \sum_{\ell=1}^{N} \left[\log(1 + \exp[-d_\ell r(\boldsymbol{z}_\ell)])\right] \tag{28}$$

Note that we hope to bound the following

$$\mathcal{L}(\widehat{r}^\star) - \mathcal{L}(r^\star), \tag{29}$$

which can be rewritten as

$$\mathcal{L}(\widehat{r}^\star) - \min_{r\in\mathcal{R}} \mathcal{L}(r) + \min_{r\in\mathcal{R}} \mathcal{L}(r) - \mathcal{L}(r^\star) \le \mathcal{L}(\widehat{r}^\star) - \min_{r\in\mathcal{R}} \mathcal{L}(r) + \nu, \tag{30}$$

by using Assumption 3.6.

Invoking Theorem 26.5 of (Shalev-Shwartz & Ben-David, 2014), we have the following generalization error bound, with probability of at least $1-\delta$:

$$\mathcal{L}(\widehat{r}^\star) - \min_{r\in\mathcal{R}} \mathcal{L}(r) \le 2\mathfrak{R}(\ell\circ\omega\circ r) + 5c\sqrt{\frac{2\ln(8/\delta)}{N}} \tag{31}$$

where the notation "$\circ$" means function composition, $\mathfrak{R}(\ell\circ\omega\circ r)$ is the Rademacher complexity of the composed function $\ell\circ\omega\circ r$, $r^\star$ is the optimal solution of (27), $\widehat{r}^\star$ the optimal solution of (28), α is an upper bound of $|r(\boldsymbol{z})|$ and $c=\log(1+e^\alpha)$, and

$$\ell(r) = \log(1+\exp(-r)), \qquad \omega(r) = dr.$$

By the properties of Rademacher complexity (Bartlett & Mendelson, 2002), we have

$$\mathfrak{R}(\ell\circ\omega\circ r) \le \mathfrak{R}_N,$$

where $\mathfrak{R}_N$ is the Rademacher complexity of r under N i.i.d. samples, since both $\ell(\cdot)$ and $\omega(\cdot)$ are 1-Lipschitz functions. Therefore, Eq. (29) becomes

$$\mathcal{L}(\widehat{r}^\star) - \mathcal{L}(r^\star) \le 2\mathfrak{R}_N + \nu + 5\log(1+e^\alpha)\sqrt{\frac{2\ln(8/\delta)}{N}}. \tag{32}$$

B.2. Bound the Distance Between the Learned Regression Function and the Optimal

Next, we will bound the following error term

$$\mathbb{E}[|\widehat{r}^\star(\boldsymbol{z}) - r^\star(\boldsymbol{z})|^2], \tag{33}$$

First, using Lemma 3.5, we have

$$\frac{\gamma_\alpha}{2}|\widehat{r}^\star - r^\star|^2 \le \ell(\boldsymbol{z}, d; \widehat{r}^\star) - \ell(\boldsymbol{z}, d; r^\star) - \langle\nabla\ell(\boldsymbol{z}, d; r^\star), \widehat{r}^\star - r^\star\rangle, \tag{34}$$

where

$$\gamma_\alpha = \frac{e^\alpha}{(1+e^\alpha)^2}.$$

Taking expectation on both sides, we have

$$\frac{\gamma_\alpha}{2}\mathbb{E}[|\widehat{r}^\star - r^\star|^2] \le \mathcal{L}(\widehat{r}^\star) - \mathcal{L}(r^\star) - \mathbb{E}\left[\langle\nabla\ell(\boldsymbol{z}, d; r^\star), \widehat{r}^\star - r^\star\rangle\right]. \tag{35}$$

We hope to show that

$$\mathbb{E}\left[\langle\nabla\ell(\boldsymbol{x}, d; r^\star), \widehat{r}^\star - r^\star\rangle\right] \ge 0.$$

Expand the left hand side, we have

$$\mathbb{E}\left[\frac{dr^\star - d\widehat{r}^\star}{1+e^{dr^\star}}\right].$$

Given the fact that

$$\mathcal{L}(\widehat{r}^\star) \ge \mathcal{L}(r^\star),$$

we have

$$\mathbb{E}\left[\log(1+e^{-d\widehat{r}^\star})\right] \geq \mathbb{E}\left[\log(1+e^{-dr^\star})\right] \Longrightarrow \mathbb{E}\left[\log\frac{1+e^{-d\widehat{r}^\star}}{1+e^{-dr^\star}}\right] \geq 0.$$

By the Jensen's inequality, we have

$$\mathbb{E}\left[\frac{1+e^{-d\widehat{r}^\star}}{1+e^{-dr^\star}}\right] \geq 1.$$

It can be re-written as

$$\mathbb{E}\left[\frac{1+e^{dr^\star}+e^{dr^\star-d\widehat{r}^\star}-1}{1+e^{dr^\star}}\right] \geq 1,$$

which is

$$\mathbb{E}\left[\frac{e^{dr^\star-d\widehat{r}^\star}-e^0}{1+e^{dr^\star}}\right] \geq 0.$$

Since e^x is monotonic, the above implies that

$$\mathbb{E}\left[\frac{dr^\star-d\widehat{r}^\star-0}{1+e^{dr^\star}}\right] \geq 0 \Rightarrow \mathbb{E}\left[\langle\nabla\ell(\boldsymbol{z},d;r^\star),\widehat{r}^\star-r^\star\rangle\right] \geq 0.$$

Thus, we have the following inequality:

$$\frac{\gamma_\alpha}{2}\mathbb{E}[|\widehat{r}^\star-r^\star|^2] \leq \mathcal{L}(\widehat{r}^\star)-\mathcal{L}(r^\star) \leq 2\mathfrak{R}_N+\nu+5c\sqrt{\frac{2\ln(8/\delta)}{N}}, \tag{36}$$

which is

$$\mathbb{E}[|\widehat{r}^\star(\boldsymbol{z})-r^\star(\boldsymbol{z})|^2] \leq \frac{(1+e^\alpha)^2}{e^\alpha}\left(2\mathfrak{R}_N+\nu+5c\sqrt{\frac{2\ln(8/\delta)}{N}}\right). \tag{37}$$

This concludes the proof. ■

C. Proof of Theorem 3.8

Let us first recall the optimal $r^\star(\boldsymbol{x},\boldsymbol{u})$'s expression in the unlimited sample case from (Hyvarinen et al., 2019). Given a two-class classification problem, where the probabilities of seeing the positive class and negative class are $p(\boldsymbol{x})$ and $q(\boldsymbol{x})=1-p(\boldsymbol{x})$, respectively. We train a classifier D using the following loss function:

$$L(D) = \mathbb{E}_{\boldsymbol{x}\sim P}\left[-\log D(\boldsymbol{x})\right] + \mathbb{E}_{\boldsymbol{x}\sim Q}\left[-\log(1-D(\boldsymbol{x}))\right] \tag{38}$$

where $D(\boldsymbol{x})$ is supposed to learn the probability of the positive class that is generated following the distribution P and $1-D(\boldsymbol{x})$ the probability of class with distribution Q.

As shown in (Goodfellow et al., 2014), the optimal solution is of the discriminator is as follows

$$D(\boldsymbol{x}) = \frac{p(\boldsymbol{x})}{p(\boldsymbol{x})+q(\boldsymbol{x})}.$$

The above can be re-written as (Hyvarinen et al., 2019):

$$\begin{aligned} D(\boldsymbol{x}) &= \frac{1}{1+q(\boldsymbol{x})/p(\boldsymbol{x})} \\ &= \frac{1}{1+\exp(-\log(p(\boldsymbol{x})/q(\boldsymbol{x})))} \end{aligned}$$

In our case, we have two classes of data, i.e., $(\boldsymbol{x}_\ell, \boldsymbol{u}_\ell) \sim P = \mathbb{P}_{\boldsymbol{x},\boldsymbol{u}}$ and $(\boldsymbol{x}_\ell, \widetilde{\boldsymbol{u}}_\ell) \sim Q = \mathbb{P}_{\boldsymbol{x}}\mathbb{P}_{\boldsymbol{u}}$. Note that $Q = \mathbb{P}_{\boldsymbol{x}}\mathbb{P}_{\boldsymbol{u}}$ because the latter was sampled from $\boldsymbol{x}$ and $\boldsymbol{u}$ randomly and independently.

Note that our discriminator is constructed as

$$D(\boldsymbol{x}, \boldsymbol{u}) = \frac{1}{1 + \exp(-r(\boldsymbol{x}, \boldsymbol{u}))},$$

following the standard logistic regression formulation. Hence, the optimal $r(\boldsymbol{x}, \boldsymbol{u})$ can be written as

$$\begin{aligned} r^\star(\boldsymbol{x}, \boldsymbol{u}) &= \log\left(p(\boldsymbol{x}, \boldsymbol{u})/p(\boldsymbol{x})p(\boldsymbol{u})\right) \\ &= \log p(\boldsymbol{x}|\boldsymbol{u}) - \log p(\boldsymbol{x}). \end{aligned} \tag{39}$$

C.1. Cross-Derivative Estimation Using Numerical Derivative

In this subsection, we estimate of the cross-derivative using the generalization bound above. First, using Lemma 3.7, we define

$$\varepsilon = \frac{(1 + e^\alpha)^2}{e^\alpha}\left(2\mathfrak{R}_N + \nu + 5c\sqrt{\frac{2\ln(8/\delta)}{N}}\right).$$

We denote

$$\varepsilon_\ell = \left(\widehat{r}^\star\left(\boldsymbol{z}_\ell\right) - r^\star\left(\boldsymbol{z}_\ell\right)\right)^2$$

as a realization of ε. Using these notations and the i.i.d. assumption, we have

$$\mathbb{E}_{\boldsymbol{z}_\ell \sim \mathcal{D}}[\varepsilon_\ell] \leq \varepsilon,$$

where $\mathcal{D}$ stands for the distribution that generates $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_N$.

Note the learned regression function $\widehat{r}^\star(\cdot)$ is constructed as follows:

$$\widehat{r}^\star(\boldsymbol{x}, \boldsymbol{u}) = \sum_{i=1}^{D} \phi_i^\star\left(h_i^\star\left(\boldsymbol{x}\right), \boldsymbol{u}\right) \tag{40}$$

where $\boldsymbol{h}^\star = (h_1^\star, \cdots, h_D^\star)$ is invertible and smooth.

Recall that the optimal regression function $r^\star(\boldsymbol{z})$ can be written as the difference of log PDF [cf. Eq. (39)]:

$$\begin{aligned} r^\star(\boldsymbol{z}) &= \log p(\boldsymbol{x}|\boldsymbol{u}) + \log p(\boldsymbol{u}) - (\log p(\boldsymbol{x}) + \log p(\boldsymbol{u})) \\ &= \left(\sum_{i=1}^{D} q_i(f_i(\boldsymbol{x}), \boldsymbol{u}) + \log p(\boldsymbol{u}) + \log\det \boldsymbol{J}\right) - (\log p_s(\boldsymbol{f}(\boldsymbol{x})) + \log p(\boldsymbol{u}) + \log\det \boldsymbol{J}) \\ &= \sum_{i=1}^{D} q_i(f_i(\boldsymbol{x}), \boldsymbol{u}) - \log p_s(\boldsymbol{f}(\boldsymbol{x})) \end{aligned} \tag{41}$$

where p_s is the distribution of $\boldsymbol{s}$, $\boldsymbol{J}$ is the Jacobian of $\boldsymbol{f} = \boldsymbol{g}^{-1}$ and the related terms are cancelled. Also recall that we have defined the relations:

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}), \quad \boldsymbol{v}(\boldsymbol{y}) = \boldsymbol{f}(\boldsymbol{h}^{-1}(\boldsymbol{y})) = \boldsymbol{s}. \tag{42}$$

Then, using (40) and (41) and the notations in (42), we have the following

$$\varepsilon_\ell = \left(\sum_{i=1}^{D} q_i(v_i(\boldsymbol{y}_\ell), \boldsymbol{u}_\ell) - \log p_s(\boldsymbol{v}(\boldsymbol{y}_\ell)) - \sum_{i=1}^{D} \phi_i\left([\boldsymbol{y}_\ell]_i, \boldsymbol{u}_\ell\right)\right)^2$$

with $\mathbb{E}_{\mathcal{D}}[\varepsilon_\ell] \leq \varepsilon$.

C.1.1. Recall the Proof of the Population Case

To understand our development, we first recall the key steps in the proof of the infinite sample case from (Hyvarinen et al., 2019).

Step 1. By training the regression function defined in (5), we have the optimal solution under unlimited infinite samples equivalent to the log PDF difference as shown in (39)

$$\underbrace{\sum_{i=1}^{D} \phi_i^\star \left(h_i^\star\left(\boldsymbol{x}\right), \boldsymbol{u}\right)}_{\widehat{r}^\star(\boldsymbol{x},\boldsymbol{u})} = \underbrace{\log p(\boldsymbol{x}|\boldsymbol{u}) - \log p(\boldsymbol{x})}_{r^\star(\boldsymbol{x},\boldsymbol{u})}. \tag{43}$$

Step 2. The equation from Step 1 can be further expanded by following (41), i.e.,

$$\sum_{i=1}^{D} \phi_i^\star\left(y_i, \boldsymbol{u}\right) = \sum_{i=1}^{D} q_i(v_i(\boldsymbol{y}), \boldsymbol{u}) - \log p_s(\boldsymbol{v}(\boldsymbol{y})). \tag{44}$$

Step 3. Denote $\eta(\boldsymbol{y}) = \log p_s(\boldsymbol{v}(\boldsymbol{y}))$. Taking derivatives w.r.t. y_j and y_k on (44), we have

$$\sum_i \left(q_i'' \frac{\partial v_i(\boldsymbol{y})}{\partial y_j}\frac{\partial v_i(\boldsymbol{y})}{\partial y_k} + q_i' \frac{\partial^2 v_i(\boldsymbol{y})}{\partial y_j \partial y_k}\right) - \frac{\partial^2 \eta(\boldsymbol{y})}{\partial y_j \partial y_k} = 0, \tag{45}$$

since the term on the LHS of Eq. (44) is gone.

By putting together

$$\boldsymbol{\kappa}_{jk} = \left[\frac{\partial v_1(\boldsymbol{y})}{\partial y_j}\frac{\partial v_i(\boldsymbol{y})}{\partial y_k}, \cdots, \frac{\partial v_1(\boldsymbol{y})}{\partial y_j}\frac{\partial v_i(\boldsymbol{y})}{\partial y_k}, \frac{\partial^2 v_1(\boldsymbol{y})}{\partial y_j \partial y_k}, \cdots, \frac{\partial^2 v_D(\boldsymbol{y})}{\partial y_j \partial y_k}\right]^\top$$

as a single vector, there could be different versions of Eq. (45) since the coefficients q_i'' and q_i' are determined by $\boldsymbol{u}$. Suppose for $\boldsymbol{u}_0$ we have

$$\sum_i \widetilde{q}_i'' \frac{\partial v_i(\boldsymbol{y})}{\partial y_j}\frac{\partial v_i(\boldsymbol{y})}{\partial y_k} + \widetilde{q}_i' \frac{\partial^2 v_i(\boldsymbol{y})}{\partial y_j \partial y_k} - \frac{\partial^2 \eta(\boldsymbol{y})}{\partial y_j \partial y_k} = 0. \tag{46}$$

Then, for another $\boldsymbol{u}_1, \cdots, \boldsymbol{u}_{2D}$ we have $2D$ different versions of (45). By subtracting (46) from the $2D$ equations and using the assumption of Variability, we have

$$\boldsymbol{W}\boldsymbol{\kappa}_{jk} = \boldsymbol{0} \tag{47}$$

where $\boldsymbol{W}$ is full column rank.

Therefore, we can reach the conclusion of Theorem 2.1 by using Fact 3.1. ∎

C.1.2. The Finite-Sample Case

Unlike the population case, one cannot directly establish the cross-derivative equations for any point $\boldsymbol{y}$. Instead, we can estimate the corresponding quantity of $\phi(\boldsymbol{y}_\ell)$ at any observed point $\boldsymbol{y}_\ell$ using numerical differentiation techniques. Similar to (44), we look at the finite sample version defined as

$$t(\boldsymbol{y}_\ell) = \sum_{i=1}^{D} q_i(v_i(\boldsymbol{y}_\ell), \boldsymbol{u}_\ell) - \log p_s(\boldsymbol{v}(\boldsymbol{y}_\ell)) - \sum_{i=1}^{D} \phi_i^\star\left([\boldsymbol{y}_\ell]_i, \boldsymbol{u}_\ell\right), \tag{48}$$

of which we hope to numerically estimate its cross-derivative, i.e., $\frac{\partial^2 t(\boldsymbol{y})}{\partial y_j \partial y_k}$. Note that under population case, $t(\boldsymbol{y}_\ell) = 0$ for any ℓ.

Next, we define:

$$\begin{aligned}
\Delta \boldsymbol{y}_{jk}^{++} &= [\mathbf{0}, \ldots, +\Delta y_j, \ldots, \mathbf{0}, \ldots, +\Delta y_k, \ldots, \mathbf{0}]^\top, \\
\Delta \boldsymbol{y}_{jk}^{+-} &= [\mathbf{0}, \ldots, +\Delta y_j, \ldots, \mathbf{0}, \ldots, -\Delta y_k, \ldots, \mathbf{0}]^\top, \\
\Delta \boldsymbol{y}_{jk}^{-+} &= [\mathbf{0}, \ldots, -\Delta y_j, \ldots, \mathbf{0}, \ldots, +\Delta y_k, \ldots, \mathbf{0}]^\top, \\
\Delta \boldsymbol{y}_{jk}^{--} &= [\mathbf{0}, \ldots, -\Delta y_j, \ldots, \mathbf{0}, \ldots, -\Delta y_k, \ldots, \mathbf{0}]^\top,
\end{aligned}$$

with $\Delta y_j > 0$ and $\Delta y_k > 0$ for any $j, k \in [D]$ with $j < k$.

Define $\boldsymbol{y}_{\widehat{\ell}} = \boldsymbol{y}_\ell + \Delta \boldsymbol{y}_{jk}^{++}$, $\boldsymbol{y}_{\widetilde{\ell}} = \boldsymbol{y}_\ell + \Delta \boldsymbol{y}_{jk}^{+-}$, $\boldsymbol{y}_{\overline{\ell}} = \boldsymbol{y}_\ell + \Delta \boldsymbol{y}_{jk}^{-+}$, and $\boldsymbol{y}_{\ell'} = \boldsymbol{y}_\ell + \Delta \boldsymbol{y}_{jk}^{--}$. Then, we have

$$\begin{aligned}
\varepsilon_{\widehat{\ell}} &= \left(\sum_{i=1}^{D} q_i(v_i(\boldsymbol{y}_{\widehat{\ell}}), \boldsymbol{u}_\ell) - \log p_s(\boldsymbol{v}(\boldsymbol{y}_{\widehat{\ell}})) - \sum_{i=1}^{D} \phi_i^\star \left([\boldsymbol{y}_{\widehat{\ell}}]_i, \boldsymbol{u}_\ell \right) \right)^2, \\
\varepsilon_{\widetilde{\ell}} &= \left(\sum_{i=1}^{D} q_i(v_i(\boldsymbol{y}_{\widetilde{\ell}}), \boldsymbol{u}_\ell) - \log p_s(\boldsymbol{v}(\boldsymbol{y}_{\widetilde{\ell}})) - \sum_{i=1}^{D} \phi_i^\star \left([\boldsymbol{y}_{\widetilde{\ell}}]_i, \boldsymbol{u}_\ell \right) \right)^2, \\
\varepsilon_{\overline{\ell}} &= \left(\sum_{i=1}^{D} q_i(v_i(\boldsymbol{y}_{\overline{\ell}}), \boldsymbol{u}_\ell) - \log p_s(\boldsymbol{v}(\boldsymbol{y}_{\overline{\ell}})) - \sum_{i=1}^{D} \phi_i^\star \left([\boldsymbol{y}_{\overline{\ell}}]_i, \boldsymbol{u}_\ell \right) \right)^2, \\
\varepsilon_{\ell'} &= \left(\sum_{i=1}^{D} q_i(v_i(\boldsymbol{y}_{\ell'}), \boldsymbol{u}_\ell) - \log p_s(\boldsymbol{v}(\boldsymbol{y}_{\ell'})) - \sum_{i=1}^{D} \phi_i^\star \left([\boldsymbol{y}_{\ell'}]_i, \boldsymbol{u}_\ell \right) \right)^2.
\end{aligned}$$

where $\boldsymbol{u}_\ell$ remains the same. Note there exist such points $\boldsymbol{z}_{\hat{\ell}} = (\boldsymbol{x}_{\hat{\ell}}, \boldsymbol{u}_\ell)$, $\boldsymbol{z}_{\widetilde{\ell}} = (\boldsymbol{x}_{\widetilde{\ell}}, \boldsymbol{u}_\ell)$, $\boldsymbol{z}_{\overline{\ell}} = (\boldsymbol{x}_{\overline{\ell}}, \boldsymbol{u}_\ell)$ and $\boldsymbol{z}_{\ell'} = (\boldsymbol{x}_{\ell'}, \boldsymbol{u}_\ell)$ in the domain of $\mathcal{X} \times \mathcal{U}$.

Using numerical differentiation of multivariate function (Mørken, 2013), the cross-derivative of a function $\psi(x, y)$ can be numerically estimated as

$$\frac{\partial^2 \psi(x,y)}{\partial x \partial y} \approx \frac{\psi(x + \Delta x, y + \Delta y) - \psi(x + \Delta x, y - \Delta y)}{4 \Delta x \Delta y} - \frac{\psi(x - \Delta x, y + \Delta y) - \psi(x - \Delta x, y - \Delta y)}{4 \Delta x \Delta y}. \tag{49}$$

The exact relation between the left and right hand sides are as follows:

$$\begin{aligned}
\frac{\partial^2 \psi(x,y)}{\partial x \partial y} = & \frac{\psi(x + \Delta x, y + \Delta y) - \psi(x + \Delta x, y - \Delta y)}{4 \Delta x \Delta y} - \frac{\psi(x - \Delta x, y + \Delta y) - \psi(x - \Delta x, y - \Delta y)}{4 \Delta x \Delta y} \\
& - \frac{\Delta x^2}{6} \frac{\partial^4 \psi(\xi_{11}, \xi_{21})}{\partial x^3 \partial y} - \frac{\Delta y^2}{6} \frac{\partial^4 \psi(\xi_{12}, \xi_{22})}{\partial x \partial y^3} - \frac{\Delta x^3}{48 \Delta y} \left(\frac{\partial^4 \psi(\xi_{13}, \xi_{23})}{\partial x^4} - \frac{\partial^4 \psi(\xi_{14}, \xi_{24})}{\partial x^4} \right) \\
& - \frac{\Delta x \Delta y}{8} \left(\frac{\partial^4 \psi(\xi_{15}, \xi_{25})}{\partial x^2 \partial y^2} - \frac{\partial^4 \psi(\xi_{16}, \xi_{26})}{\partial x^2 \partial y^2} \right) - \frac{\Delta y^3}{48 \Delta x} \left(\frac{\partial^4 \psi(\xi_{17}, \xi_{27})}{\partial y^4} - \frac{\partial^4 \psi(\xi_{18}, \xi_{28})}{\partial y^4} \right),
\end{aligned}$$

where $\xi_{1i} \in (x - \Delta x, x + \Delta x)$ and $\xi_{2i} \in (y - \Delta y, y + \Delta y)$ for $i \in \{1, \cdots, 8\}$.

Denote $\eta(\boldsymbol{y}_\ell) = \log p_s(\boldsymbol{v}(\boldsymbol{y}_\ell))$. Note that the analytical form of cross derivative $\frac{\partial^2 t(\boldsymbol{y}_\ell)}{\partial y_j \partial y_k}$ is

$$\frac{\partial^2 t(\boldsymbol{y}_\ell)}{\partial y_j \partial y_k} = \sum_i q_i'' \frac{\partial v_i(\boldsymbol{y}_\ell)}{\partial y_j} \frac{\partial v_i(\boldsymbol{y}_\ell)}{\partial y_k} + q_i' \frac{\partial^2 v_i(\boldsymbol{y}_\ell)}{\partial y_j \partial y_k} - \frac{\partial^2 \eta(\boldsymbol{y}_\ell)}{\partial y_j \partial y_k}, \tag{50}$$

which can be also expressed as

$$\begin{aligned}
\frac{\partial^2 t(\boldsymbol{y}_\ell)}{\partial y_j \partial y_k} = & \frac{\pm \sqrt{\varepsilon_{\widehat{\ell}}} \mp \sqrt{\varepsilon_{\widetilde{\ell}}} \mp \sqrt{\varepsilon_{\overline{\ell}}} \pm \sqrt{\varepsilon_{\ell'}}}{4 \Delta y_j \Delta y_k} - \frac{\Delta y_j^2}{6} \frac{\partial^4 t(\boldsymbol{\xi}_1)}{\partial y_j^3 \partial y_k} - \frac{\Delta y_k^2}{6} \frac{\partial^4 t(\boldsymbol{\xi}_2)}{\partial y_j \partial y_k^3} - \frac{\Delta y_j^3}{48 \Delta y_k} \left(\frac{\partial^4 t(\boldsymbol{\xi}_3)}{\partial y_j^4} - \frac{\partial^4 t(\boldsymbol{\xi}_4)}{\partial y_j^4} \right) \\
& - \frac{\Delta y_j \Delta y_k}{8} \left(\frac{\partial^4 t(\boldsymbol{\xi}_5)}{\partial y_j^2 \partial y_k^2} - \frac{\partial^4 t(\boldsymbol{\xi}_6)}{\partial y_j^2 \partial y_k^2} \right) - \frac{\Delta y_k^3}{48 \Delta y_j} \left(\frac{\partial^4 t(\boldsymbol{\xi}_7)}{\partial y_k^4} - \frac{\partial^4 t(\boldsymbol{\xi}_8)}{\partial y_k^4} \right),
\end{aligned} \tag{51}$$

where $\boldsymbol{\xi}_m$'s are vectors satisfying

$$\boldsymbol{\xi}_m = \theta_m \boldsymbol{y}_{\widehat{\ell}} + (1-\theta_m)\boldsymbol{y}_{\ell'},\ m \in \{1, \cdots, 8\},$$

where $\theta_m \in (0,1)$.

Note that with a different $\boldsymbol{u}_{\widetilde{\ell}}$, we have the following

$$\begin{aligned}
&\sum_i \widetilde{q}_i'' \frac{\partial v_i(\boldsymbol{y}_\ell)}{\partial y_j}\frac{\partial v_i(\boldsymbol{y}_\ell)}{\partial y_k} + \widetilde{q}_i' \frac{\partial^2 v_i(\boldsymbol{y}_\ell)}{\partial y_j \partial y_k} - \frac{\partial^2 \eta(\boldsymbol{y}_\ell)}{\partial y_j \partial y_k} \\
&= \frac{\pm\sqrt{\widetilde{\varepsilon}_{\widehat{\ell}}} \mp \sqrt{\widetilde{\varepsilon}_{\widetilde{\ell}}} \mp \sqrt{\widetilde{\varepsilon}_{\widetilde{\ell}}} \pm \sqrt{\widetilde{\varepsilon}_{\ell'}}}{4\Delta y_j \Delta y_k} - \frac{\Delta y_j^2}{6}\frac{\partial^4 t(\widetilde{\boldsymbol{\xi}}_1)}{\partial y_j^3 \partial y_k} - \frac{\Delta y_k^2}{6}\frac{\partial^4 t(\widetilde{\boldsymbol{\xi}}_2)}{\partial y_j \partial y_k^3} - \frac{\Delta y_j^3}{48\Delta y_k}\left(\frac{\partial^4 t(\widetilde{\boldsymbol{\xi}}_3)}{\partial y_j^4} - \frac{\partial^4 t(\widetilde{\boldsymbol{\xi}}_4)}{\partial y_j^4}\right) \\
&- \frac{\Delta y_j \Delta y_k}{8}\left(\frac{\partial^4 t(\widetilde{\boldsymbol{\xi}}_5)}{\partial y_j^2 \partial y_k^2} - \frac{\partial^4 t(\widetilde{\boldsymbol{\xi}}_6)}{\partial y_j^2 \partial y_k^2}\right) - \frac{\Delta y_k^3}{48\Delta y_j}\left(\frac{\partial^4 t(\widetilde{\boldsymbol{\xi}}_7)}{\partial y_k^4} - \frac{\partial^4 t(\widetilde{\boldsymbol{\xi}}_8)}{\partial y_k^4}\right),
\end{aligned} \tag{52}$$

By subtracting (52) from (51), the $\frac{\partial^2 \eta(\boldsymbol{y}_\ell)}{\partial y_j \partial y_k}$ term is gone. Taking absolute value and expectation w.r.t $\boldsymbol{y}$

$$\begin{aligned}
&\mathbb{E}\left[\left|\sum_i (q_i'' - \widetilde{q}_i'')\frac{\partial v_i(\boldsymbol{y}_\ell)}{\partial y_j}\frac{\partial v_i(\boldsymbol{y}_\ell)}{\partial y_k} + (q_i' - \widetilde{q}_i')\frac{\partial^2 v_i(\boldsymbol{y}_\ell)}{\partial y_j \partial y_k}\right|\right] \\
&\leq \frac{\mathbb{E}\left[\sqrt{\varepsilon_{\widehat{\ell}}}\right] + \mathbb{E}\left[\sqrt{\varepsilon_{\widetilde{\ell}}}\right] + \mathbb{E}\left[\sqrt{\varepsilon_{\widetilde{\ell}}}\right] + \mathbb{E}\left[\sqrt{\varepsilon_{\ell'}}\right]}{4\Delta y_j \Delta y_k} + \frac{\mathbb{E}\left[\sqrt{\widetilde{\varepsilon}_{\widehat{\ell}}}\right] + \mathbb{E}\left[\sqrt{\widetilde{\varepsilon}_{\widetilde{\ell}}}\right] + \mathbb{E}\left[\sqrt{\widetilde{\varepsilon}_{\widetilde{\ell}}}\right] + \mathbb{E}\left[\sqrt{\widetilde{\varepsilon}_{\ell'}}\right]}{4\Delta y_j \Delta y_k} \\
&\quad + \frac{\Delta y_j^2}{3}C_t + \frac{\Delta y_k^2}{3}C_t + \frac{\Delta y_j^3 C_t}{12\Delta y_k} + \frac{\Delta y_j \Delta y_k C_t}{2} + \frac{\Delta y_k^3 C_t}{12\Delta y_j} \\
&\leq \frac{2\sqrt{\varepsilon}}{\Delta y_j \Delta y_k} + \frac{\Delta y_j^2}{3}C_t + \frac{\Delta y_k^2}{3}C_t + \frac{\Delta y_j^3 C_t}{12\Delta y_k} + \frac{\Delta y_j \Delta y_k C_t}{2} + \frac{\Delta y_k^3 C_t}{12\Delta y_j}
\end{aligned}$$

where the first inequality is by triangle inequality and the assumption on the bound of fourth-order derivative, while the second inequality is by Jensen's inequality, i.e. $\sqrt{\mathbb{E}[x]} \geq \mathbb{E}[\sqrt{x}]$.

Then, we hope to find the optimal upper bound

$$\inf_{\Delta y_j, \Delta y_k} \frac{2\sqrt{\varepsilon}}{\Delta y_j \Delta y_k} + \frac{\Delta y_j^2}{3}C_t + \frac{\Delta y_k^2}{3}C_t + \frac{\Delta y_j^3 C_t}{12\Delta y_k} + \frac{\Delta y_j \Delta y_k C_t}{2} + \frac{\Delta y_k^3 C_t}{12\Delta y_j}.$$

To find an upper bound, we let $\Delta y = \Delta y_j = \Delta y_k$, with $\Delta y > 0$. Such simplification leads to a looser upper bound but easier to derive. Then, we have the following optimization problem:

$$\inf_{\Delta y} \frac{2\sqrt{\varepsilon}}{\Delta y^2} + \frac{\Delta y^2}{3}C_t + \frac{\Delta y^2}{3}C_t + \frac{\Delta y^2 C_t}{12} + \frac{\Delta y^2 C_t}{2} + \frac{\Delta y^2 C_t}{12}. \tag{53}$$

Note that the function in (53) is convex, so the optimal is at

$$\Delta y^\star = \left(\frac{36\sqrt{\varepsilon}}{C_t}\right)^{1/4}.$$

Then, the cross-derivative can be bounded by

$$\mathbb{E}\left[\left|\sum_i (q_i'' - \widetilde{q}_i'')\frac{\partial v_i(\boldsymbol{y}_\ell)}{\partial y_j}\frac{\partial v_i(\boldsymbol{y}_\ell)}{\partial y_k} + (q_i' - \widetilde{q}_i')\frac{\partial^2 v_i(\boldsymbol{y}_\ell)}{\partial y_j \partial y_k}\right|\right] \leq \frac{\sqrt{3C_t}\varepsilon^{1/4}}{3}.$$

With

$$\varepsilon = \frac{(1+e^{\alpha})^2}{e^{\alpha}}\left(2\mathfrak{R}_N + \nu + 5\log(1+e^{\alpha})\sqrt{\frac{2\ln(8/\delta)}{N}}\right),$$

we have

$$\mathbb{E}\left[\left|\sum_i (q_i'' - \widetilde{q}_i'')\frac{\partial v_i(\boldsymbol{y}_\ell)}{\partial y_j}\frac{\partial v_i(\boldsymbol{y}_\ell)}{\partial y_k} + (q_i' - \widetilde{q}_i')\frac{\partial^2 v_i(\boldsymbol{y}_\ell)}{\partial y_j \partial y_k}\right|\right] \leq \frac{\sqrt{3C_t}(1+e^{\alpha})^{1/2}}{3e^{\alpha/4}}\left(2\mathfrak{R}_N + \nu + 5\log(1+e^{\alpha})\sqrt{\frac{2\ln(8/\delta)}{N}}\right)^{1/4}, \tag{54}$$

for all pairs of (j,k) with $j<k$. Thus we have $D(D-1)/2$ such inequalities above.

By the assumption of Variability (Hyvarinen et al., 2019), there exists $\boldsymbol{u}_i$ with $i \in \{0,\cdots,2D\}$, such that the matrix

$$\boldsymbol{W} = [\boldsymbol{w}(\boldsymbol{y},\boldsymbol{u}_1) - \boldsymbol{w}(\boldsymbol{y},\boldsymbol{u}_0),\cdots,\boldsymbol{w}(\boldsymbol{y},\boldsymbol{u}_{2D}) - \boldsymbol{w}(\boldsymbol{y},\boldsymbol{u}_0)],$$

as defined in (9) is full rank where $\boldsymbol{w}(\boldsymbol{y},\boldsymbol{u})$ is defined (10). This further implies that we have $2D$ different versions of (54) with various coefficients. Putting the $2D$ inequalities together, we have the following bound

$$\mathbb{E}\left[\|\boldsymbol{W}\boldsymbol{\kappa}_{jk}\|_1\right] \leq 2D\frac{\sqrt{3C_t}(1+e^{\alpha})^{1/2}}{3e^{\alpha/4}}\left(2\mathfrak{R}_N + \nu + 5\log(1+e^{\alpha})\sqrt{\frac{2\ln(8/\delta)}{N}}\right)^{1/4}, \tag{55}$$

with the vector $\boldsymbol{\kappa}_{jk}$ defined earlier

$$\boldsymbol{\kappa}_{jk} = \left[\frac{\partial v_1(\boldsymbol{y}_\ell)}{\partial y_j}\frac{\partial v_1(\boldsymbol{y}_\ell)}{\partial y_k},\cdots,\frac{\partial v_D(\boldsymbol{y}_\ell)}{\partial y_j}\frac{\partial v_D(\boldsymbol{y}_\ell)}{\partial y_k},\underbrace{\frac{\partial^2 v_1(\boldsymbol{y}_\ell)}{\partial y_j \partial y_k},\cdots,\frac{\partial^2 v_D(\boldsymbol{y}_\ell)}{\partial y_j \partial y_k}}_{\boldsymbol{\gamma}_{jk}^{\top}}\right]^{\top}.$$

Therefore, for any (j,k) pair, we have

$$\mathbb{E}\left[\|\boldsymbol{\gamma}_{jk}\|_2\right] \leq \mathbb{E}\left[\|\boldsymbol{\kappa}_{jk}\|_2\right] \leq \frac{2D\sqrt{3C_t}(1+e^{\alpha})^{1/2}}{3e^{\alpha/4}\sigma_*^2}\left(2\mathfrak{R}_N + \nu + 5\log(1+e^{\alpha})\sqrt{\frac{2\ln(8/\delta)}{N}}\right)^{1/4},$$

where we use the inequality $\|\boldsymbol{x}\|_2 \leq \|\boldsymbol{x}\|_1$ and $\sigma_* = \max_{\boldsymbol{W}} \sigma_{\min}(\boldsymbol{W})$. This completes the proof. ■