AN ABSTRACT OF THE THESIS OF

Vladyslav Pauk for the degree of Master of Science in Computer Science presented on November 13, 2024.

Title: Post-Nonlinear Mixture Identification via Variational Auto-Encoding

Abstract approved: ______________________________

Xiao Fu

Simplex component analysis, which estimates latent variables constrained to a probability simplex, plays a critical role in applications such as brain signal classification, speech separation, remote sensing, and causal discovery. These applications often involve data that exhibit nonlinear relationships, which cannot be effectively captured by traditional linear mixture models. Recent approaches address these limitations by employing autoencoder-based architectures with structured latent spaces, extending linear models to nonlinear domains. However, these methods often struggle in noisy settings and lack the ability to uniquely identify latent components without additional linear unmixing. This thesis introduces a probabilistic deep generative framework based on variational Bayesian inference, which facilitates unique identification of latent components even in noisy nonlinear environments. Building on the variational autoencoder framework for linear probabilistic simplex component analysis, the proposed model is designed to address post-nonlinear distortions by leveraging neural networks. The thesis presents a rigorous theoretical analysis of the identifiability conditions in probabilistic post-nonlinear simplex component analysis and validates these theoretical guarantees through systematic numerical simulations.

Post-Nonlinear Mixture Identification via Variational Auto-Encoding

by

Vladyslav Pauk

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented November 13, 2024
Commencement June 2025

Master of Science thesis of Vladyslav Pauk presented on November 13, 2024.

APPROVED:

Major Professor, representing Computer Science

Head of the School of Electrical Engineering and Computer Science

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Vladyslav Pauk, Author

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

Chapter 1: Introduction

> *"I thought of a labyrinth of labyrinths, of one sinuous spreading labyrinth that would encompass the past and the future and in some way involve the stars."*
>
> Jorge Luis Borges, The Garden of Forking Paths

Over the past decade, *artificial intelligence* (AI) has rapidly advanced, gaining widespread adoption in both research and industry and fundamentally transforming the modern technological and scientific landscape[1]. At the heart of modern AI systems are neural networks, a class of parametric models with universal approximation capabilities inspired by the neural processing in animal brains. Rooted in foundational research on backpropagation [38, 48] and stochastic gradient-based optimization [8], neural networks enable data-driven modeling in various domains, from natural language processing and computer vision to autonomous systems and signal processing.

The proliferation of digital data from sources such as the internet, mobile devices, IoT sensors, and social media has generated vast datasets essential for training complex neural networks. High-performance computing hardware, particularly Graphics Processing Units (GPUs) and, more recently, Tensor Processing Units (TPUs), has played a pivotal role in managing massive datasets and complex computations with efficiency. Additionally, the advent of distributed and cloud computing has democratized access to large-scale training resources, enabling researchers and developers to train models that would otherwise be constrained by local hardware limitations. These technological advances, coupled with development of new algorithmic architectures, such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and attention-based Transformers, have unlocked the full potential of *deep learning*. Deep learning leverages Artificial Neural Networks (ANNs), complex parametric models with universal approximation capabilities inspired by biological neural systems, to model real-world data with remarkable accuracy. This approach has led to ground-breaking achievements across diverse applications, including healthcare, finance, science, robotics, and entertainment among others.

Machine learning is a computational approach that enables systems to learn from examples by identifying underlying patterns within data. In a nutshell, training a model entails iteratively adjusting its parameters based on training examples to minimize the discrepancy between the model's output and the ground truth. Also known as *supervised learning*, this paradigm relies on labeled datasets where the input-output relationships are explicitly defined. A key limitation of supervised learning is its dependence on often manually labeled data, which can be both scarce and costly to obtain, hindering its scalability and utility in many applications.

In contrast, *unsupervised learning* focuses on uncovering latent structures and complex patterns within unlabeled input data by relying on intrinsic relationships between the data points. This approach is most notably applied in *density estimation* and *generative modeling*, giving

[1] For a comprehensive and recent overview of AI advancements, adoption trends, and their societal impacts, see [43].

rise to celebrated models like Variational Autoencoders (VAEs) [35], Generative Adversarial Networks [22], and Sstable Diffusion [47], where algorithms learn to model the underlying data probability distribution. These models revolutionized the field of generative AI, due to their ability to generate new samples that closely resemble the input data distribution, producing highly realistic synthetic data with remarkable fidelity. Unsupervised learning techniques are particularly effective in representation learning, where they are employed to transform high-dimensional data into more compact and interpretable forms, while capturing its underlying structure in a model-independent way. However, unsupervised learning is inherently challenging. The lack of labeled data means that the evaluation and interpretation of learned representations and the identification of meaningful patterns require more advanced algorithms and techniques. These methods often require intricate design and optimization, making unsupervised learning a sophisticated and computationally intensive process.

Finding meaningful low-dimensional representations for multivariate data is one of central challenges in machine learning, particularly within the domain of *dimensionality reduction*. Latent Variable Models (LVMs) address this challenge by assuming that the data is generated by an unknown intrinsic process that depends on a small number of hidden (or latent) variables. For example, in many applications, high-dimensional data is generated through a mixing process, where the observations are represented as a linear combination of lower-dimensional latent variables or factors. This class of models is known as Linear Mixture Models (LMM). The Unsupervised Mixture Learning (UML) encompasses a variety of methods also known in the literature as Blind Source Separation and Factor Analysis [25] that aim to recover the latent components from the observed data with the limited prior knowledge of the mixing process or the characteristics of the latent components. UML methods are essential for tasks like audio and speech separation [20], EEG signal denoising [36], image representation learning [2], hyperspectral unmixing [5], and topic mining [18], among others.

An essential aspect of UML is *identifiability*, which pertains to whether the latent components in a model can be uniquely and reliably identified without the need for labeled data or explicit training examples. Identifiability provides a rigorous foundation for meaningful parameter interpretation in LVM. It is well-known, that linear mixture models with Gaussian prior are generally not identifiable because multiple solutions can exist without additional constraints [28]. Incorporation of structural assumptions on the latent variables, such as statistical independence, nonnegativity, boundedness, sparsity, and simplex structures, form the basis for widely adopted models like Independent Component Analysis (ICA) [25], Nonnegative Matrix Factorization [17], Bounded Component Analysis [13], Sparse Component Analysis [57], and Simplex-Structured Matrix Factorization (SSMF) [20, 18]. By leveraging such structural constraints, these UML frameworks can not only render the problem more tractable but also provide meaningful interpretations of the latent components, enhancing their applicability across various domains.

The structural assumptions often arise naturally from the physical or domain-specific characteristics of the underlying problem. For instance, the simplex structure, central to this work, is commonly encountered in compositional data, representing proportions of a whole. Specifically, each data point corresponds to a categorical probability distribution, represented as a vector with non-negative elements that sum to one. This structure is particularly relevant in applications such as topic mining, hyperspectral unmixing [5], community detection [24], and image representation learning [56], among other.

The LMMs, while effective in many applications, have limitations. In real-world scenarios, the observed data often undergoes unknown nonlinear distortions, which can significantly affect the performance of linear UML methods. This is observed in various applications, including hyperspectral imaging, audio processing, wireless communications, and brain imaging [3, 58]. The nonlinear character of the problem presents significant challenges, particularly when compared to the well-established linear UML setting. As shown in [27, 29], even strong assumptions like statistical independence of the latent components are insufficient to guarantee identifiability in nonlinear mixture models.

The identifiability of latent components in the presence of unknown nonlinear transformations was elusive until recently, when it has gained renewed interest, due to its connection to unsupervised deep learning [27, 29]. In [29], the authors addressed this challenge by approaching it through the lens of nonlinear Independent Component Analysis (nICA), and provided the first nonlinear identifiability guarantees for deep latent-variable models, offering a rigorous framework for recovering independent latent variables under nonlinear mixture model. In many practical scenarios, the assumption of statistical independence between latent components is often overly restrictive, limiting the applicability of ICA models. While some efforts have extended LMMs to handle specific types of nonlinear distortions, such as bi-linear, linear-quadratic, and polynomial transformations [7, 16, 14], addressing general nonlinear mixtures without resorting to statistical independence of the latent factors remains a significant challenge.

The post-nonlinear (PNL) mixture model [52, 6] offers a nonlinear extension to the linear mixture model (LMM) by introducing component-wise nonlinear transformations following the linear mixing process. This model has been effectively applied to complex tasks in signal processing and data analytics, including nonlinear hyperspectral unmixing, image embedding, brain signal classification, speech separation, remote sensing, causal discovery, and nonlinear clustering. A notable advancement in [55] provided rigorous nonlinear identification criteria for the PNL model, demonstrating that unknown post-nonlinear distortions can be removed under specific geometric assumptions on the latent components. These models are based on the premise that the latent components reside in a lower-dimensional subspace, which imposes interdependencies between the nonlinear transformations. The simplex-constrained post-nonlinear mixture (SC-PNM) model, proposed in [55, 41], leverages the assumption that latent components are generated from the probability simplex. Further, [42] demonstrated that under mild conditions, the presence of a nontrivial null space in the underlying mixing system is sufficient to guarantee the identification and removal of unknown nonlinearities.

However, while these methods successfully eliminate nonlinear distortions, they do not fully recover the latent components, requiring additional constraints and linear methods for component extraction. Moreover, since these methods tackle nonlinear distortions in a deterministic manner, they exhibit performance deterioration in the presence of significant noise. Additionally, the use of geometric constraints via constrained optimization introduces computational complexity, as repeated optimization steps are required, affecting scalability of the method.

Probabilistic methods, such as Bayesian inference, are essential for addressing noise in data. In hyperspectral unmixing (HU), Bayesian inference has been applied to manage uncertainty, but the resulting intractable integrals in certain probability density functions often necessitate computationally expensive techniques like Markov chain Monte Carlo (MCMC) sampling [15]. Deep generative models, such as VAE, have gained popularity for their ability to model complex data distributions and generate realistic synthetic data. In these models, data is modeled as a probabilistic function of hidden latent variables, which are sampled from a prior distribution, and the generative process is learned through the gradient-based optimization of the maximum likelihood objective.

In [32], the problem of nonlinear identifiability was tackled by embedding nonlinear ICA into the VAE framework, extending it to handle noisy or incomplete observations within a maximum-likelihood context. This work demonstrated that, for a broad class of deep latent variable models, the true joint distribution of observed and latent variables can be identified, up to simple transformations, enabling robust disentanglement in noisy nonlinear settings. This approach requires a factorized prior distribution over the latent variables, conditioned on an additional observed variable (e.g., class label), and is akin to nonlinear multiview analysis [40], where multiple data views It also connects to identifiable flow-based generative models, which emerge as a special case. This approach relies on a prior factorized over the latent variables, and conditioned on an auxiliary observed variable (e.g., class label), along the lines of nonlinear multiview analysis [40], and contrast learning [32]. Moreover, it connects to identifiable flow-based generative models, which emerge as a special case of this framework.

One advantage of probabilistic models is their ability to naturally incorporate the geometric constraints into the model architecture, by choosing an appropriate prior distribution. Building on this idea, PRISM employs the maximum likelihood inference for noisy linear mixture model with the simplex-distributed prior, and provides linear identifiability guarantees up to a permutation ambiguity. Probabilistic Simplex Component Analysis (PRISM) addresses the intractability of the likelihood function through approximation methods such as importance sampling and variational inference approximation (VIA), which can be computationally expensive for large-scale problems. Expanding on PRISM, VAE-based simplex component analysis (VASCA) addresses optimization of the maximum likelihood objective through the use of a VAE, modeling the variational posterior with ANNs. This enhances the model's expressive power, bypasses computational bottlenecks, and retains the probabilistic structure and identifiability guarantees of the original framework.

This thesis aims to extend existing research by addressing the limitations of deterministic nonlinear unsupervised mixture learning models through a reformulation within a probabilistic framework, to study nonlinear identifiability in deep generative models on the example of nonlinear simplex component analysis (nSCA). Unlike deterministic approaches, the probabilistic formulation captures both global and local geometric structures of the data, enabling simultaneous removal of nonlinearity and latent component disentanglement. This eliminates the need for pipelining with linear methods or imposing additional structural constraints. The proposed framework offers several practical advantages, including robustness to noise and generative capabilities, making it applicable to a variety of tasks. Specific applications include portfolio analysis, ECG signal processing, and hyperspectral unmixing, where accurate separation of latent components from noisy, mixed data is crucial for improved decision-making and analysis. By adopting a probabilistic approach, the model is not only suitable for mapping complex data into interpretable forms, but also for scientific and financial simulations, where it can generate realistic nonlinear synthetic data.

As a first step in this agenda, this thesis focuses on the nonlinear simplex component analysis (nSCA) model, where latent components are generated from the probability simplex. Building on the linear VASCA framework, this approach modifies the architecture to handle nonlinear distortions and address optimization challenges. We introduce a nonlinear decoder and redesign the encoder to more effectively capture the relationships between the linearly mixed components. The central contribution of this work is analysis of the nonlinear identifiability of the probabilistic noisy PNL model in nSCA setting, and the development of a scalable latent identification inference algorithm with provable identifiability guarantees. We provide a detailed description of the theoretical foundation and implementation of the algorithm, formulate and prove identifiability of the model, and evaluate it in experiments on synthetic data.

The following manuscript is organized as follows. In Chapter 2, we provide a formal introduction to the theoretical foundations of probability density estimation, variational inference, variational autoencoders, and the concept of identifiability. Chapter 4 presents the core model of this work, alongside the primary theoretical contributions. In Chapter 5, we conduct experiments on synthetic datasets, where we evaluate and analyze the model's performance under various settings. Following the experiments, we discuss the results, highlighting key insights, challenges, and potential avenues for future research. Notations, supplementary reference materials, and technical derivations are provided in the Appendix.

Chapter 2: Theoretical Background

2.1 Notations and Definitions

Our notation follows standard conventions in linear algebra and probability theory. The set of real numbers is denoted by $\mathbb{R}$, with $\mathbb{R}_+$ and $\mathbb{R}_{++}$ representing the sets of non-negative and positive numbers, respectively. Euclidean vectors are represented by boldfaced lowercase letters, such as $\boldsymbol{x}$. For a vector $\boldsymbol{x} \in \mathbb{R}^N$, its components are denoted by indexed lowercase letters x_n, where $\boldsymbol{n} = 1, \ldots, N$. Matrices are denoted by uppercase letters, for example, A. For a matrix $A \in \mathbb{R}^{N\times M}$, its columns are written as $\boldsymbol{a}_m \in \mathbb{R}^N$, for $m = 1, \ldots, M$, i.e. $A = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_M]$. Special matrices and vectors include the identity matrix $\boldsymbol{I}$, the all-zero vector $\mathbf{0}$, and the all-one vector $\mathbf{1}$. The diagonal matrix with elements x_n is written as $\operatorname{diag}(\boldsymbol{x})$, and the Euclidean norm of a vector is denoted by $\|\boldsymbol{x}\|$. Operators $\operatorname{tr}(A)$, $\operatorname{rank}(A)$, and $\operatorname{krank}(A)$ denote the trace, rank, and the Kruskal rank of a matrix, respectively. Notations $A^\top$, A^{-1}, and A^+ refer to the transpose, inverse, and pseudo-inverse of A, respectively. A matrix A is orthogonal if $A^\top A = \boldsymbol{I}$. For a matrix $A \in \mathbb{R}^{N\times M}$ with full column rank, the left pseudo-inverse is explicitly defined as $A^+ = (A^\top A)^{-1}A^\top$. A matrix $A \in \mathbb{R}^{N\times M}$ is affinely independent, if $\overline{A} = [\boldsymbol{a}_1 - \boldsymbol{a}_M, \ldots, \boldsymbol{a}_{M-1} - \boldsymbol{a}_M]$ has full column rank. The set of all affinely independent matrices in $\mathbb{R}^{N\times M}$ is denoted by $\mathcal{A}^{N\times M}$. The indicator function for a set $\mathcal{X}$ is defined as:

$$\mathbf{1}_{\mathcal{X}}(\boldsymbol{x}) = \begin{cases} 1, & \text{if } \boldsymbol{x} \in \mathcal{X}, \\ 0, & \text{otherwise,} \end{cases}$$

The binary matrix operators are defined as follows. The Kronecker product, denoted by $\otimes$, for $A \in \mathbb{R}^{N\times M}$ and $B \in \mathbb{R}^{P\times Q}$, is given by:

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1M}B \\ \vdots & \ddots & \vdots \\ a_{N1}B & \cdots & a_{NM}B \end{bmatrix}.$$

The Hadamard product, denoted by $\odot$, for $A, B \in \mathbb{R}^{N\times M}$, is defined as:

$$A \odot B = \begin{bmatrix} a_{11}b_{11} & \cdots & a_{1M}b_{1M} \\ \vdots & \ddots & \vdots \\ a_{N1}b_{N1} & \cdots & a_{NM}b_{NM} \end{bmatrix}.$$

Finally, the Khatri-Rao product, denoted by $\circledast$, for $A \in \mathbb{R}^{N\times K}$ and $B \in \mathbb{R}^{M\times K}$, is expressed as:

$$A \circledast B = \begin{bmatrix} \boldsymbol{a}_1 \otimes \boldsymbol{b}_1 & \cdots & \boldsymbol{a}_K \otimes \boldsymbol{b}_K \end{bmatrix}.$$

To describe geometric structures, we use the span, affine hull, and open convex hull of the columns of a matrix $A \in \mathbb{R}^{N \times M}$, defined, respectively, as:

$$\begin{aligned}\operatorname{span}(A) &= \{\boldsymbol{x} = A\boldsymbol{z} \mid \boldsymbol{z} \in \mathbb{R}^M\},\\ \operatorname{aff}(A) &= \{\boldsymbol{x} = A\boldsymbol{z} \mid \boldsymbol{z} \in \mathbb{R}^M,\ \mathbf{1}^\top \boldsymbol{z} = 1\},\\ \operatorname{conv}(A) &= \{\boldsymbol{x} = A\boldsymbol{z} \mid \boldsymbol{z} \in \mathbb{R}^M_+,\ \mathbf{1}^\top \boldsymbol{z} = 1\}.\end{aligned}$$

The null space of A is defined as the set of all vectors that are mapped to zero by A:

$$\operatorname{null}(A) = \{\boldsymbol{z} \in \mathbb{R}^M \mid A\boldsymbol{z} = \mathbf{0}\}.$$

A simplex is defined as the convex hull of an affinely independent set of points, referred to as its vertices. Let $A = [\boldsymbol{a}_1, \dots, \boldsymbol{a}_M]$ where $\boldsymbol{a}_m \in \mathbb{R}^N$ represent the vertices of a $(M-1)$-simplex. A simplex is full-dimensional in $\mathbb{R}^N$ if $M = N+1$, meaning it is spanned by $N+1$ points in N-dimensional space. The volume of a full-dimensional simplex is given by $\operatorname{vol}(A) = \frac{1}{N!}|\det \overline{A}|$. In general, when the simplex resides in a lower-dimensional subspace of $\mathbb{R}^N$, i.e. $M \leq N$, the volume can be recast in terms of the Gram determinant:

$$\operatorname{vol}(A) = \frac{\left(\det \overline{A}^\top \overline{A}\right)^{1/2}}{(M-1)!}. \tag{2.1}$$

The unit simplex and its open counterpart are respectively defined as

$$\begin{aligned}\Delta^{M-1} &= \{\boldsymbol{z} \in \mathbb{R}^M_+ \mid \mathbf{1}^\top \boldsymbol{z} = 1\},\\ \bar{\Delta}^{M-1} &= \{\boldsymbol{z} \in \mathbb{R}^M_{++} \mid \mathbf{1}^\top \boldsymbol{z} = 1\}.\end{aligned}$$

The Lebesgue integral over the unit simplex can be expressed using coordinates $\bar{\boldsymbol{z}} = [z_1, \dots, z_{M-1}]$ as follows:

$$\int\limits_{\Delta^{M-1}} f(\boldsymbol{z})\, d\boldsymbol{\mu}(\boldsymbol{z}) = \int\limits_{\mathbb{R}^{M-1}_+} f(\bar{\boldsymbol{z}}, 1 - \mathbf{1}^\top \bar{\boldsymbol{z}})\, d\bar{\boldsymbol{z}}, \tag{2.2}$$

where $z_M = 1 - \mathbf{1}^\top \bar{\boldsymbol{z}}$ ensuring that $\boldsymbol{z}$ satisfies the simplex constraint.

We assume all functions are real-valued and measurable. The composition of functions is denoted by $(f \circ g)(x) = f(g(x))$. The continuous p-times differentiable functions are denoted by C^p, and the Lebesgue-integrable functions by L^1. The Fourier transform of a function f is denoted as $\mathcal{F}[f](\boldsymbol{\omega})$, where $\boldsymbol{\omega}$ is the frequency variable. We also use softmax function, that is defined as

$$\operatorname{softmax}(\boldsymbol{z})_n = \frac{e^{z_n}}{\sum_{m=1}^M e^{z_m}},$$

where $\boldsymbol{z} \in \mathbb{R}^M$. Finally, the multivariate beta-function is

$$B(\boldsymbol{x}) = \frac{\prod_{n=1}^N \Gamma(x_n)}{\Gamma(\sum_{n=1}^N x_n)},$$

where $\boldsymbol{x} \in \mathbb{R}^N$ and $\Gamma(\cdot)$ is the gamma function. The Dirac delta function $\delta(\boldsymbol{x}$ is defined by

$$\int f(\boldsymbol{x})\delta(\boldsymbol{x}-\boldsymbol{z})\,d\boldsymbol{x} = f(\boldsymbol{z}).$$

The calligraphic uppercase letters $\mathcal{N}$, $\mathcal{D}$, and $\mathcal{LN}$ denote the normal, Dirichlet, and logistic-normal probability distributions or their families depending on the context, respectively. A random variable $\boldsymbol{x}$ distributed according to $\mathcal{N}$ is denoted by $\boldsymbol{x} \sim \mathcal{N}$. The expectation of $\boldsymbol{x}$ with distribution p is denoted by

$$\mathbb{E}_{\boldsymbol{x}\sim p}[f(\boldsymbol{x})] = \int\limits_{\mathcal{X}} f(\boldsymbol{x})p(\boldsymbol{x})d\boldsymbol{x},$$

where $f(\cdot) \in L^1$, and $\mathcal{X}$ is the support of the distribution p. The variance and covariance of $\boldsymbol{x}$ are denoted by $\mathrm{var}(\boldsymbol{x})$ and $\mathrm{cov}(\boldsymbol{x})$, respectively.

2.2 Probability Density Estimation

Estimating the probability density of data is a fundamental problem in statistical modeling and unsupervised machine learning. It serves as a critical tool for understanding data distributions, facilitating Bayesian inference by enabling the computation of posterior distributions from observed data. Moreover, density estimation is a cornerstone of generative modeling, underpinning frameworks such as VAE, normalizing flows, and other probabilistic models. Beyond generative tasks, it supports key applications such as clustering, anomaly detection, and dimensionality reduction, making it indispensable for exploratory data analysis and the development of robust machine learning systems.

Broadly, probability density estimation methods fall into two categories: non-parametric and parametric. Non-parametric approaches, including histograms, kernel density estimation, and nonparametric regression, approximate data distributions without assuming a fixed functional form for the probability density. Instead, they infer the form of the distribution from data, making them highly flexible for modeling complex structures. However, as dataset size grows, these methods require an increasing number of parameters, often resulting in infinite-dimensional parametric space. Along with the dimensionality curse, this scalability issue often leads to overfitting and computational inefficiency, limiting the applicability of non-parametric methods, especially for high-dimensional data. Moreover, the absence of prior knowledge limits the interpretability of non-parametric methods and constrains their utility in generative modeling tasks, where structured assumptions about the underlying distribution are often essential.

In contrast, parametric methods assume that the data follows a specific distribution family, characterized by a fixed number of parameters estimated from the data. For example, the Gaussian Mixture Model (GMM) [1] describes the data distribution as a combination of multiple independent Gaussian components, with the parameters optimized to maximize the likelihood of the data. Parametric methods are generally computationally efficient and interpretable, as their form is fixed regardless of the dataset size and stems from domain knowledge. However, due to inherent assumptions about the distribution structure, they often lack the flexibility to capture realistic data distributions. Recent advances in deep learning have extended the scope of parametric models. By leveraging neural networks to parameterize distributions, techniques such as VAE and GAN enabled modeling of intricate nonlinear structures in data in a flexible and expressive manner, while maintaining computational efficiency, and have become indispensable for density estimation in high-dimensional and complex datasets.

Maximum likelihood estimation (MLE) is a foundational approach to parameter estimation in statistical modeling, and serves as the basis for many density estimation techniques. The likelihood function is a measure of how well the model explains empirical data, and is defined as the joint probability density of the data viewed as a function of the model parameters. Instead of using the likelihood itself, MLE maximizes the log-likelihood. As a monotonically increasing function, the logarithm does not affect the optimal solution, but it transforms products into sums, simplifying calculations, improving numerical stability and avoiding underflow. Given a dataset of T independent and identically distributed samples $\boldsymbol{x}^{(t)} \sim p^*(\boldsymbol{x})$, the empirical log-likelihood function is:

$$L_T(\boldsymbol{\theta}) = \sum_{t=1}^{T} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(t)}), \tag{2.3}$$

The idea of MLE is to maximize the likelihood function, which implies that the observed data is most probable under the specified statistical model:

$$\hat{\boldsymbol{\theta}}_{\mathrm{ML}} = \arg\max_{\boldsymbol{\theta}} L_T(\boldsymbol{\theta}). \tag{2.4}$$

MLE provides a consistent estimate of the data distribution under general conditions.

Statistical distances provide a measure of the dissimilarity or divergence between probability distributions. In the context of the density estimation, minimizing the statistical distance between the true distribution $p^*(\boldsymbol{x})$ and its parametric approximation $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ provides an estimator of the underlying data distribution $p^*(\boldsymbol{x})$, a method known as minimum distance estimation. A commonly used statistical distance, the Kullback-Leibler (KL) divergence measures the expected logarithmic difference between two distributions:

$$D_{KL}(p^* \| p_{\boldsymbol{\theta}}) = \mathbb{E}_{\boldsymbol{x} \sim p^*} \left[\log \frac{p^*(\boldsymbol{x})}{p_{\boldsymbol{\theta}}(\boldsymbol{x})} \right]. \tag{2.5}$$

Also known as relative entropy, KL-divergence quantifies the expected information loss incurred by using the approximate distribution rather than the true distribution:

$$D_{\mathrm{KL}}(p^* \| p_{\boldsymbol{\theta}}) = H(p^*, p_{\boldsymbol{\theta}}) - H(p^*),$$

where $H(p^*) = -\mathbb{E}_{\boldsymbol{x} \sim p^*}[\log p^*(\boldsymbol{x})]$ is the entropy of p^* representing the information content of a true distribution, and $H(p^*, p_{\boldsymbol{\theta}}) = -\mathbb{E}_{\boldsymbol{x} \sim p^*}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x})]$ is the cross entropy of $p_{\boldsymbol{\theta}}$ relative to p^* quantifying the uncertainty in approximating the true distribution with the model distribution.

By the law of large numbers, the empirical log-likelihood converges to the expected log-likelihood in the population limit:

$$\lim_{T\to\infty} L_T(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x}\sim p^*}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x})] := L(\boldsymbol{\theta}).$$

Maximizing the log-likelihood, thus minimizes the distance, in terms of KL divergence, between the approximate distribution $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ and the true distribution $p^*(\boldsymbol{x})$, effectively aligning them. By minimizing the KL divergence between $p^*(\boldsymbol{x})$ and $p_\theta(\boldsymbol{x})$, ML identifies the closest approximation, even when the true distribution is not within the model family. This ensures that, in practical applications, the estimated parameters offer the best possible fit within the constraints of the model, even if the model is imperfect and the data is finite.

2.3 Variational Inference

2.3.1 Latent Variable Models

Latent variable models (LVMs) describe observed data by assuming an underlying generative process driven by a set of unobserved factors, known as latent variables. A well-known example of LVMs is the factor analysis model, which assumes that the data $\boldsymbol{x}$ is generated by a linear transformation A of the latent variables $\boldsymbol{z}$, followed by the addition of Gaussian noise $\boldsymbol{v}$:

$$\boldsymbol{x} = A\boldsymbol{z} + \boldsymbol{v}.$$

By incorporating prior knowledge about the data-generating process—such as assumptions about the distribution of latent variables or the relationships between variables—LVMs can better capture patterns and dependencies in the data. In addition, LVMs allow construction of complex distributions by combining simpler components. For example, GMMs approximate non-Gaussian distributions by mixing multiple Gaussian components, guided by discrete latent variables representing cluster memberships.

Formally, a probabilistic LVM is defined as the joint distribution of observed and latent variables:

$$p_\theta(\boldsymbol{x}, \boldsymbol{z}) = p_\theta(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z}), \tag{2.6}$$

where $p(\boldsymbol{z})$ is the prior probability of the latent variables, and $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ is the conditional probability of the observed data given $\boldsymbol{z}$ (see Figure 2.1). Learning about the latent variable $\boldsymbol{z}$ involves inferring the posterior distribution $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$, which represents the probability of latent states given observed data. This conditional distribution can be used to estimate the latent variables, and construct predictive data densities, among other tasks. The posterior distribution is defined by Bayes' theorem:

$$p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}) = \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{p_{\boldsymbol{\theta}}(\boldsymbol{x})}, \tag{2.7}$$

where $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ is the marginal probability, or the evidence. It is calculated by marginalizing (2.6) over the latent variables:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z}. \tag{2.8}$$

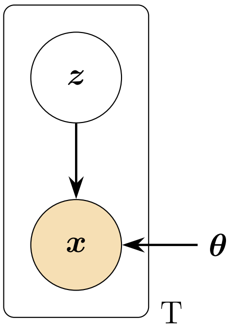

Figure 2.1: Ditected graphical model representation of a latent variable model (LVM). The observed data is represented by the shaded node $\boldsymbol{x}$, and the latent variables by the unshaded node $\boldsymbol{z}$. Solid lines denote the generative model $p(z)p_\theta(x|z)$, with the generative model parameters $\boldsymbol{\theta}$.

Except for very specific forms of $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})$ and $p(\boldsymbol{z})$, this marginalization is in general not analytically tractable, and often suffers from the curse of dimensionality [9], thus requiring approximate inference methods.

Markov Chain Monte Carlo (MCMC) has long been a standard framework for approximate inference in Bayesian settings. The method constructs an ergodic Markov chain over the latent variables, such that its stationary distribution aligns with the target posterior. Samples are then drawn from the chain and used to form an empirical approximation of the posterior. When dealing with large datasets, traditional MCMC methods face scalability issues due to the necessity of evaluating the likelihood across the entire dataset at each iteration. This requirement leads to substantial computational overhead, making MCMC impractical for big data applications.

2.3.2 Evidence-Lower Bound

Variational inference (VI) is an optimization-based framework for approximating intractable probability distributions, widely used in machine learning and Bayesian modeling [30]. Unlike MCMC methods, VI does not rely on sampling, and offers a fast and scalable alternative for large and complex datasets. By transforming the inference problem into an optimization task, VI seeks the optimal member from a tractable family of distributions $q_\phi(\boldsymbol{z})$ with variational parameters $\phi \in \Phi$, that closely approximates the true posterior, as measured by the Kullback-Leibler (KL) divergence:

$$\hat{q}(\boldsymbol{z}) = \arg\min_{\phi \in \Phi} D_{KL}(q_\phi(\boldsymbol{z}) \,\|\, p_\theta(\boldsymbol{z}|\boldsymbol{x})).$$

The KL divergence in the optimization problem above cannot be computed directly due to the intractability of the posterior $p_\theta(\boldsymbol{z}|\boldsymbol{x})$. However, the problem can be recast into a computationally tractable form by reframing it as the maximization of a lower bound on the marginal likelihood, commonly referred to as the evidence lower bound (ELBO).

To derive the ELBO, we first rewrite the logarithm of evidence as an expectation over the latent variables $\boldsymbol{z}$ using the variational density $q_\phi(\boldsymbol{z})$:

$$\log p_\theta(\boldsymbol{x}) = \log \mathbb{E}_{\boldsymbol{z} \sim q_\phi(\boldsymbol{z})} \left[\frac{p_\theta(\boldsymbol{x}, \boldsymbol{z})}{q_\phi(\boldsymbol{z})} \right].$$

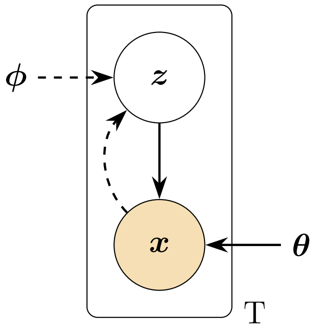

Figure 2.2: Graphical model representation of the variational inference approximation (VIA). Solid lines denote the generative model $p_\theta(z)p_\theta(x|z)$, dashed lines denote the inference model, represented by the variational approximation $q_\phi(z|x)$ to the intractable posterior $p_\theta(z|x)$. The variational parameters ϕ are learned jointly with the generative model parameters θ.

Applying Jensen's inequality, we obtain the lower bound on the evidence:

$$\log p_\theta(\boldsymbol{x}) \geq \mathbb{E}_{\boldsymbol{z}\sim q_\phi(\boldsymbol{z})}\left[\log \frac{p_\theta(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z})}\right],$$

where the right-hand side defines the ELBO objective:

$$\ell(\boldsymbol{\theta},\boldsymbol{\phi};\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{z}\sim q_\phi(\boldsymbol{z})}\left[\log \frac{p_\theta(\boldsymbol{x},\boldsymbol{z})}{q_\phi(\boldsymbol{z})}\right].$$

The difference between the log-evidence and the ELBO is the KL divergence between the variational and the true posteriors:

$$\log p_\theta(\boldsymbol{x}) = \ell(\boldsymbol{\theta},\boldsymbol{\phi};\boldsymbol{x}) + D_{KL}(q_{\boldsymbol{\phi}}(\cdot)\,\|\,p_\theta(\cdot|\boldsymbol{x})). \tag{2.9}$$

When $q_{\boldsymbol{\phi}}(\boldsymbol{z}) = p_\theta(\boldsymbol{z}|\boldsymbol{x})$, the KL term vanishes, and the ELBO is equal to the evidence. Hence, the ELBO provides a tractable surrogate for the intractable marginal likelihood, enabling an optimization-based reformulation of MLE in (2.3):

$$\hat{\boldsymbol{\theta}}_{ELBO}, \hat{\boldsymbol{\phi}}_{ELBO} = \max_{\boldsymbol{\theta},\boldsymbol{\phi}} \frac{1}{T}\sum_{t=1}^{T} \ell(\boldsymbol{\theta},\boldsymbol{\phi};\boldsymbol{x}^{(t)}). \tag{2.10}$$

Due to linearity of expectations ELBO exhibits different linear decompositions. By explicitly balancing the trade-offs within the loss function, one can adjust the model to emphasize different properties of the learned representation. A commonly used form is expressed as a difference between a reconstruction term and a KL divergence with respect to the prior $p(\boldsymbol{z})$:

$$\ell(\boldsymbol{\theta},\boldsymbol{\phi};\boldsymbol{x}) = \mathbb{E}_{\boldsymbol{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] - D_{KL}(q_{\boldsymbol{\phi}}\,\|\,p). \tag{2.11}$$

Maximizing ELBO simultaneously attempts to keep $q_{\boldsymbol{\phi}}(\cdot|\boldsymbol{x})$ close to $p(\boldsymbol{z})$ and concentrate $q_{\boldsymbol{\phi}}(\cdot|\boldsymbol{x})$ on those $\boldsymbol{z}$ that likely generated $\boldsymbol{x}$. An alternative form of the variational objective emphasizes the role of entropy in posterior approximation:

$$\ell(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) = \mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) + h_{\phi}(\boldsymbol{z})]. \tag{2.12}$$

where we omit the constant prior term $\log p(\boldsymbol{z})$, and $h_{\phi}(\boldsymbol{z}) = -\log q_{\boldsymbol{\phi}}(\boldsymbol{z})$ is the point-wise differential entropy of the variational distribution. The entropy term $h_{\phi}(\boldsymbol{z})$ encourages the variational posterior $q_{\phi}(\boldsymbol{z})$ to maintain a broader distribution, mitigating overfitting by preventing the posterior from collapsing into an overly narrow representation. By promoting exploration, this term ensures that the model captures a wider range of plausible latent values, thus supporting more robust posterior approximations. Additionally, entropy serves as a measure of uncertainty in the variational posterior. High entropy reflects greater uncertainty in the latent representation, while low entropy indicates a more confidence in posterior approximation. This property is particularly useful for model diagnostics and gaining insights into how uncertainty is distributed within the learned representation.

2.3.3 Variational Autoencoder

Variational Autoencoders (VAEs) [35] are a class of generative models that integrate variational Bayesian inference with deep learning. These models parameterize the conditional distributions by neural networks, optimized using stochastic gradient descent and backpropagation. VAEs employ the autoencoder architecture, consisting of an encoder network $\phi(\boldsymbol{x})$ that maps data points $\boldsymbol{x}$ to a posterior parameterization $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$, and a decoder network $\theta(\boldsymbol{x})$ that maps latent variables sampled from the posterior back to the data space.

Optimization with gradient-based methods requires differentiation of expectations over $\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ with respect to the variational parameters $\boldsymbol{\phi}$. The naive Monte Carlo estimators, however, suffer from high variance, rendering them impractical for optimization (see, e.g., [46]). To address this, a differentiable estimator can be constructed by reparameterizing the latent variable $\boldsymbol{z}$ as a deterministic transformation of an auxiliary variable $\boldsymbol{\epsilon}$ that is independent of the variational parameters $\boldsymbol{\phi}$, also known as the reparameterization trick [35]. Specifically, we reparameterize $\boldsymbol{z}$ as $\boldsymbol{z} = g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}, \boldsymbol{x})$, where $\boldsymbol{\epsilon}$ is drawn from a fixed distribution $p(\boldsymbol{\epsilon})$. This allows us to rewrite the original expectation as:

$$\mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[f(\boldsymbol{z})] = \int q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) f(\boldsymbol{z})\, d\boldsymbol{z} = \int p(\boldsymbol{\epsilon}) f(g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}, \boldsymbol{x}))\, d\boldsymbol{\epsilon}.$$

Since $\boldsymbol{\epsilon}$ is independent of $\boldsymbol{\phi}$, the function $f(g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}, \boldsymbol{x}))$ is differentiable with respect to $\boldsymbol{\phi}$. We can thus approximate the expectation using Monte Carlo sampling as:

$$\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[f(\boldsymbol{z})] \approx \frac{1}{R} \sum_{l=1}^{R} f(g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}^{(r)}, \boldsymbol{x})) \quad \text{where } \boldsymbol{\epsilon}^{(r)} \sim p(\boldsymbol{\epsilon}),$$

which provides a differentiable estimator that can be optimized using gradient-based methods.

As an example, consider the univariate Gaussian prior distribution $z \sim \mathcal{N}(\mu, \sigma)$. We can reparameterize the latent variable as $z = \mu + \sigma^{1/2}\epsilon$, where $\epsilon \sim \mathcal{N}(0, 1)$, so that the expectation becomes:

$$\mathbb{E}_{z\sim\mathcal{N}(\mu,\sigma)}[f(z)] = \mathbb{E}_{\epsilon\sim\mathcal{N}(0,1)}[f(\mu + \sigma^{1/2}\epsilon)].$$

Using Monte Carlo sampling, we can approximate this expectation as:

$$\mathbb{E}_{z\sim\mathcal{N}(\mu,\sigma^2)}[f(z)] \approx \frac{1}{R}\sum_{l=1}^{R} f(\mu + \sigma^{1/2}\epsilon^{(r)}) \quad \text{where } \epsilon^{(r)} \sim \mathcal{N}(0, 1),$$

which is now differentiable with respect to both μ and σ.

The reparameterization trick is not limited to Gaussian distributions. It can be applied to any distribution where a suitable transformation of the form $\boldsymbol{z} = g_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}, \boldsymbol{x})$ can be found. For example, for distributions in the location-scale family (such as the Laplace or Student's t-distribution), we can reparameterize using the standard form of the distribution as the noise variable $\boldsymbol{\epsilon}$. Additionally, for some distributions, such as the log-normal or gamma distributions, reparameterization can be achieved through composition of simpler transformations. In cases where reparameterization is not straightforward, approximate methods such as inverse CDF transformations or numerical approximations can be employed to achieve similar results. This flexibility makes the reparameterization trick a powerful tool for constructing differentiable Monte Carlo estimators, enabling the efficient optimization of complex models involving stochastic components.

By applying reparameterization technique to the ELBO (2.11) we obtain the stochastic gradient variational Bayes (SGVB) estimator:

$$\tilde{\ell}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}^{(t)}) = \frac{1}{R}\sum_{r=1}^{R} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(t)}, \boldsymbol{z}^{(t,r)}) - \log q_{\boldsymbol{\phi}}(\boldsymbol{z}^{(t,r)}|\boldsymbol{x}^{(t)}), \tag{2.13}$$

where $\boldsymbol{z}^{(t,r)} = g_{\phi}(\boldsymbol{\epsilon}^{(t,r)}, \boldsymbol{x}^{(t)})$ and $\boldsymbol{\epsilon}^{(t,r)} \sim p(\boldsymbol{\epsilon})$. This formulation facilitates computation of gradients with respect to both the generative model parameters $\boldsymbol{\theta}$ and the variational parameters $\boldsymbol{\phi}$, enabling the use of stochastic gradient descent for optimization.

Next, we discuss the architecture of a VAE and its components on the example fo the AEVB variant of the VAE [35]. In this setup, both the prior $p_{\boldsymbol{\theta}}(\boldsymbol{z})$ and the posterior $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ are Gaussian, enabling closed-form computation of the KL divergence. The prior over the latent variables is defined as a centered isotropic multivariate Gaussian, $p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{0}, I)$. The conditional likelihood $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ is modeled as a multivariate Gaussian for real-valued data or Bernoulli for binary data, with the distribution parameters computed via an MLP. The approximate posterior $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ is assumed to follow a multivariate Gaussian with diagonal covariance. The log of the approximate posterior is:

$$\log q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) = \log \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2\boldsymbol{I}),$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are outputs of an encoding MLP, parameterized by $\boldsymbol{\phi}$ and conditioned on $\boldsymbol{x}$. For

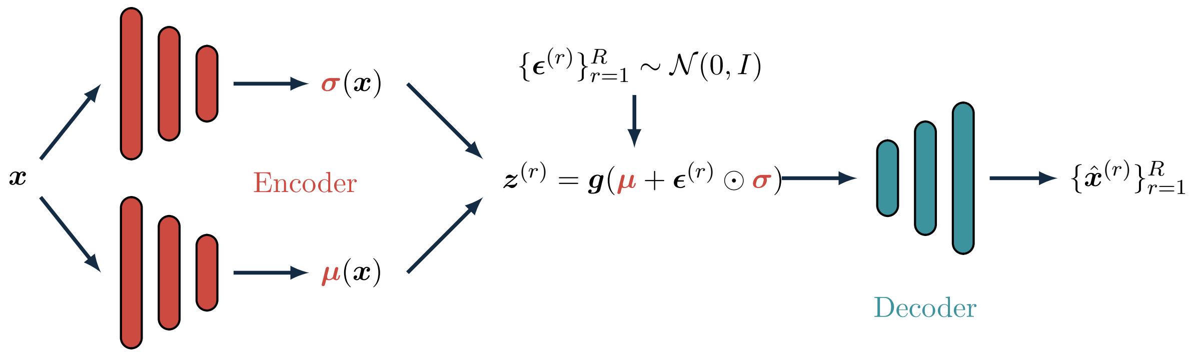
$\boldsymbol{x}$
$\boldsymbol{\sigma}(\boldsymbol{x})$
Encoder
$\boldsymbol{\mu}(\boldsymbol{x})$
$\{\boldsymbol{\epsilon}^{(r)}\}_{r=1}^{R} \sim \mathcal{N}(0, I)$
$\boldsymbol{z}^{(r)} = \boldsymbol{g}(\boldsymbol{\mu} + \boldsymbol{\epsilon}^{(r)} \odot \boldsymbol{\sigma})$
Decoder
$\{\hat{\boldsymbol{x}}^{(r)}\}_{r=1}^{R}$

Figure 2.3: Architecture of a variational autoencoder (VAE). The input $\boldsymbol{x}$ is processed by an encoder represented by stacked blue layers denoting the neural netwrorks. The encoder outputs the mean $\boldsymbol{\mu}$ and standard deviation $\boldsymbol{\sigma}$, defining a Gaussian distribution. The reparameterization trick $\boldsymbol{z}^{(r)} = \boldsymbol{\mu} + \boldsymbol{\epsilon}^{(r)} \odot \boldsymbol{\sigma}$, where $\boldsymbol{\epsilon}^{(r)} \sim \mathcal{N}(0, I)$, allows for sampling of the latent variable $\boldsymbol{z}^{(r)}$. The sampled latent variable then flows through the decoder layers, reconstructing the input as a sample $\{\hat{\boldsymbol{x}}^{(r)}\}_{r=1}^{R}$.

a datapoint $\boldsymbol{x}$, the variational lower bound becomes (we omit the sample index for brevity):

$$\tilde{\ell}_{AEVB}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) = \frac{1}{2}\sum_{m=1}^{M}\left[1 + \log(\sigma_m^2) - \mu_m^2 - \sigma_m^2\right] + \frac{1}{R}\sum_{r=1}^{R}\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}^{(r)}),$$

where $\boldsymbol{z}^{(r)} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}^{(r)}$, and $\boldsymbol{\epsilon}^{(r)} \sim \mathcal{N}(\mathbf{0}, I)$. The architecture of a AEVB algorithm is illustrated in Figure 2.3.

2.4 Simplex Component Ananlysis

2.4.1 Probability Simplex

The probability simplex is a geometric construct representing the space of probability distributions over a finite set of mutually exclusive events, or categories. Each point in a probability $(M-1)$-simplex corresponds to a unique distribution across M categories, and each coordinate represents the probability of a particular category, such that the probabilities across all categories sum to one, and thus belong to the unit simplex Δ^{M-1}. The Dirichlet distribution is a multivariate generalization of the Beta distribution, and is commonly used for modeling random variables on the unit simplex. The $\boldsymbol{\alpha}$-Dirichlet distribution with concentration parameter $\boldsymbol{\alpha} \in \mathbb{R}_{+}^{M}$ has the probability density function:

$$\mathcal{D}(\boldsymbol{z}; \boldsymbol{\alpha}) = \frac{\mathbb{1}_{\overline{\Delta}}(\boldsymbol{z})}{B(\boldsymbol{\alpha})}\prod_{i=1}^{M} z_i^{\alpha_i - 1},$$

where parameter vector $\boldsymbol{\alpha}$ dictates the distribution's shape. When $\boldsymbol{\alpha} = \mathbf{1}$, the Dirichlet distribution reduces to the uniform distribution over the $(M-1)$-simplex, which is a common choice for the prior in simplex component analysis models:

$$\mathcal{D}(\boldsymbol{z}; \mathbf{1}) = (M-1)!\,\mathbb{1}_{\overline{\Delta}^{M-1}}(\boldsymbol{z}) \tag{2.14}$$

More details on the Dirichlet distribution and its properties are provided in Appendix A.

Another important simplex distribution is the logistic-normal, or $\mathcal{LN}$-distribution, obtained by transforming the multivariate normal distribution, which can be framed as a location-scale family distribution. Multivariate location-scale distributions describe random variables whose transformation is distributed according to a multivariate normal distribution with the diagonal covariance matrix. This can be expressed as the following random variable model:

$$\boldsymbol{z} = g(\boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}), \tag{2.15}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$, and $\boldsymbol{g}$ is a differentiable transformation. The general form of the location-scale family is given by:

$$p_{\boldsymbol{g},\boldsymbol{\sigma},\boldsymbol{\mu}}(\boldsymbol{z}) = \prod_{m=1}^{M} p(z_m) = J_g^{-1} \prod_{m=1}^{M} \frac{1}{\sigma_m} \phi\left(\frac{g^{-1}(z_m) - \mu_m}{\sigma_m} \right), \tag{2.16}$$

where $\phi(\cdot)$ is the standard normal distribution, μ_m and σ_m are the location and scale parameters, respectively, $g(\cdot)$ is the transformation and J_g is the Jacobian of the transformation. The location-scale family provides a flexible and expressive parametric distribution that allows for easy sampling and stochastic optimization. The $\mathcal{LN}$-distribution is obtained by applying the additive logistic transformation $g(\boldsymbol{z}) = \text{softmax}([\boldsymbol{z}, 0])$ to a centered Gaussian distribution with the diagonal covariance, or equivalently as the probability distribution of a random variable whose multinomial logit is a normal distribution. The probability density function of the $\mathcal{LN}$-distribution is:

$$\mathcal{LN}(\boldsymbol{z}; \boldsymbol{\mu}, \boldsymbol{\sigma}) = \frac{1}{|2\pi \operatorname{diag}(\boldsymbol{\sigma})|^{1/2}} \left(\prod_{m=1}^{M} z_m \right)^{-1} \exp\left(-\frac{1}{2} \tilde{\boldsymbol{z}}^\top \operatorname{diag}(\boldsymbol{\sigma})^{-1} \tilde{\boldsymbol{z}} \right), \tag{2.17}$$

where

$$\tilde{\boldsymbol{z}} = \log\left(\frac{\boldsymbol{z}_{-M}}{z_M} \right) - \boldsymbol{\mu},$$

and $\boldsymbol{z}_{-M} = [z_1, \ldots, z_{M-1}]$ and $z_M = 1 - \sum_{m=1}^{M-1} z_m$, the location parameter is given by the Gaussian mean and the diagonal scale matrix is the covariance matrix. The $\mathcal{LN}$-distribution is commonly used as a posterior parameterization in simplex component analysis models, as it provides a flexible and tractable distribution over the probability simplex.

2.4.2 Probabilistic Simplex Component Analysis

The Probabilistic Simplex Component Analysis (PRISM) [54] addresses the problem of identifying simplex vertices from a noisy set of data points, where each observation $\boldsymbol{x} \in \mathbb{R}^N$ is modeled as:

$$\boldsymbol{x} = A\boldsymbol{z} + \boldsymbol{v}. \tag{2.18}$$

Here, $\boldsymbol{z} \in \Delta^{M-1}$ represents 1-Dirichlet latent variables, and $\boldsymbol{v}$ is Gaussian noise with zero mean and diagonal covariance $\sigma^2 I$. The vertex matrix of the simplex $A \in \mathcal{A}^{N\times M}$, has affinely independent columns, ensuring a well-defined simplex structure.

The objective in PRISM is to estimate the vertices of the simplex through ML inference, given a set of observations $\{\boldsymbol{x}^{(t)} \,|\, t = 1, \ldots, T\}$:

$$\hat{A} \in \underset{A\in\mathbb{R}^{N\times M}}{\arg\max}\, \ell(A) := \frac{1}{T}\sum_{t=1}^{T} \log p_A(\boldsymbol{x}^{(t)}), \tag{2.19}$$

where $p_A(\boldsymbol{x})$ is the likelihood of an observation under parameter A. For model (2.18), the likelihood is given by the Lebesgue integral of the Gaussian distribution over the simplex:

$$p_A(\boldsymbol{x}) = (M-1)! \int \phi_\sigma(\boldsymbol{x} - A\boldsymbol{z})\, 1_{\bar{\Delta}}(\boldsymbol{z})\, d\mu(\boldsymbol{z}),$$

where $\phi_\sigma(\boldsymbol{x})$ is the centered isotropic multivariate Gaussian distribution with variance σ^2, $\phi_\sigma(\boldsymbol{x}) = (2\pi\sigma^2)^{-N/2} \exp\left(-1/2\sigma^2 \|\boldsymbol{x}\|^2\right)$.

The integral in the likelihood expression lacks a closed-form solution, making direct ML inference challenging. One possible alternative is to maximize the joint likelihood over both A and the latent variables $\boldsymbol{z}^{(t)}$, for $t = 1, \ldots, T$. This leads to a Simplex-Structured Matrix Factorization (SSMF) problem:

$$\min_{A\in\mathbb{R}^{N\times M}, Z\in\Delta^T} \|X - AZ\|^2,$$

where $X = [\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(T)}]$ denotes the matrix of observed data points, and $Z = [\boldsymbol{z}_1, \ldots, \boldsymbol{z}_T]$ is a matrix containing simplex-constrained latent variables. However, SSMF suffers from an identifiability issue: if R is any invertible matrix in Δ^N, then (AR^{-1}, RZ) is also a solution to the problem. This lack of uniqueness implies that even noise-free solutions may not recover the true vertices accurately.

$$\max_{A\in\mathcal{A}} L_T(A) = \frac{1}{T}\sum_{t=1}^{T} \log p_A(A\boldsymbol{z}^{(t)}).$$

In the noiseless case, the log likelihood simplifies to:

$$\log p_A(\boldsymbol{x}) = -\log \mathrm{vol}(A) + \log(1_{\mathrm{conv}(A)}(\boldsymbol{x})),$$

where

$$\log(1_{\mathrm{conv}(A)}(\boldsymbol{x})) = \begin{cases} 0, & \boldsymbol{x} \in \mathrm{conv}(A) \\ -\infty, & \boldsymbol{x} \notin \mathrm{conv}(A). \end{cases}$$

Thus, the ML problem reduces to:

$$\min_{A\in\mathcal{A}} \log \mathrm{vol}(A) \quad \text{s.t.} \quad \boldsymbol{x}^{(t)} \in \mathrm{conv}(A), \quad t = 1, \ldots, T.$$

This objective seeks the minimum-volume simplex $\text{conv}(A)$ that encloses all data points $\boldsymbol{x}^{(t)}$, aligning PRISM with the simplex volume minimization [20, 19] approach in the noiseless case.

PRISM frames VI approximation by restricting the family of all $\bar{\Delta}$-supported distributions to the Dirichlet family $\mathcal{D}(\boldsymbol{\alpha})$, where $\boldsymbol{\alpha} \in \mathbb{R}^N_{++}$. This restriction yields a lower-bound approximation of the ML objective,

$$\hat{\ell}(A, \boldsymbol{\alpha}; \boldsymbol{x}) = \mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{\alpha}}}[\log(p_A(\boldsymbol{x}, \boldsymbol{z})/q_{\boldsymbol{\alpha}}(\boldsymbol{z}))]$$

motivated by two factors. Under $q_{\boldsymbol{\alpha}} = \mathcal{D}(\alpha)$, the function $\hat{\ell}(A, q; \boldsymbol{x})$ in (2.4.2) simplifies to

$$-\hat{\ell}(A, \alpha; \boldsymbol{x}) \propto \frac{1}{2\sigma^2}\mathbb{E}[\|\boldsymbol{x} - A\boldsymbol{z}\|^2] - H_{\boldsymbol{\alpha}}(\boldsymbol{z}) = \frac{1}{2\sigma^2}\left(\|\boldsymbol{x} - A\mathbb{E}[\boldsymbol{z}]\|^2 + \text{tvar}(A\boldsymbol{z})\right) - H(\boldsymbol{z}), \quad (2.20)$$

where we denote $\mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{\alpha}}}[\cdot] = \mathbb{E}[\cdot]$ for brevity; $H_{\boldsymbol{\alpha}}(\boldsymbol{z}) = \mathbb{E}[-\log q_{\boldsymbol{\alpha}}(\boldsymbol{z})]$ is the entropy; and $\text{tvar}(x) = \sum_{n=1}^N \text{var}(x_n) = \text{tr}(\text{cov}(\boldsymbol{x}))$. All $\mathbb{E}[\boldsymbol{z}]$, $\text{cov}(\boldsymbol{z})$, and $H_{bm\alpha}(\boldsymbol{z})$ have explicit expressions, making the ML problem under the Dirichlet restriction, or VIA-ML, tractable.

The structure of the Dirichlet distribution enables analytical expression of expectations and resulting in a lightweight algorithm. This structure simplifies inference, but the Dirichlet assumption is often restrictive for broader applications. For more flexible distributions, the Importance Sampling Approximation (ISA) can be applied to approximate $\hat{\ell}(A, \sigma; \boldsymbol{x}^{(t)})$:

$$\hat{\ell}(A, \sigma; \boldsymbol{x}^{(t)}) \approx \frac{1}{R}\sum_{r=1}^{R}\left(\log p_A(\boldsymbol{x}^{(t)}, \boldsymbol{z}^{(t,r)}) - \log q_\sigma^{(t)}(\boldsymbol{z}^{(t,r)})\right), \quad (2.21)$$

where $\{\boldsymbol{z}^{(t,r)}\}_{r=1}^R$ are samples drawn from the distribution $q_\sigma^{(t)}$, typically chosen as $q_\sigma^{(t)} \propto \phi_\sigma(\boldsymbol{x}^{(t)} - A\boldsymbol{z}^{(t)})\, 1_\Delta(\boldsymbol{z}^{(t)})$. Sampling can be implemented via rejection or Markov Chain Monte Carlo (MCMC) methods [49, 54]. However, generating effective samples under the structural constraints of SCA poses significant challenges. For instance, in rejection sampling over 99.9% of samples may be discarded, when $M > 10$ [54], rendering this method impractical for moderate M.

2.4.3 Variational Simplex Component Analysis

The VAE-based Simplex Component Analysis (VASCA) approaches probabilistic SCA problem by framing it within VAE framework. It employs the variational ML framework by utilizing the $\mathcal{LN}$-distribution as a variational posterior, and leveraging neural networks to model its parameters. As an $\mathcal{LN}$-distribution belongs to the location-scale family, it permits a straightforward reparameterization trick, which enables efficient, gradient-based optimization. This setup integrates the required probabilistic structure for SCA while allowing for expressive and scalable posterior approximations.

Specifically, the logistic-normal variational posterior $q_{\boldsymbol{\phi}} \in \mathcal{LN}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is parameterized by the neural networks $\boldsymbol{\mu}(\boldsymbol{x})$ and $\boldsymbol{\sigma}(\boldsymbol{x})$, where $\boldsymbol{\Sigma}(\boldsymbol{x}) = \text{Diag}(\boldsymbol{\sigma}(\boldsymbol{x})^2)$. Given (2.6) and (2.17), the SGVB estimator (2.13) can be written as

$$\tilde{\ell}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) = \frac{1}{R} \sum_{r=1}^{R} \left\{ \log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}^{(r)}) + \log p(\boldsymbol{z}^{(r)}) - \log q_{\boldsymbol{\phi}}(\boldsymbol{z}^{(r)}|\boldsymbol{x}) \right\}, \tag{2.22}$$

where $\boldsymbol{z}^{(r)} = g(\boldsymbol{\mu}(\boldsymbol{x}) + \boldsymbol{\sigma}(\boldsymbol{x}) \odot \boldsymbol{\epsilon}^{(r)})$, and $\boldsymbol{\epsilon}^{(r)} \sim \mathcal{N}(0, \boldsymbol{I})$. The uniform prior (2.14) yields a constant $\log p(\boldsymbol{z}) = \log \Gamma(M)$.

Substituting these expressions back into equation (2.22) yields expression for the optimization objective:

$$\tilde{\ell}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) = \frac{1}{R} \sum_{r=1}^{R} \left[-\frac{1}{2\sigma^2} \ell_{rec} + h_{\phi}(\boldsymbol{z}^{(r)}; \boldsymbol{x}) \right] + C, \tag{2.23}$$

where the individual terms are given by:

$$\begin{aligned} h_{\phi}(\boldsymbol{z}; \boldsymbol{x}) &= \frac{1}{2} \tilde{z}^{\top} \operatorname{Diag}(\boldsymbol{\sigma}(\boldsymbol{x}))^{-1} \tilde{z} + \frac{1}{2} \sum_{i=1}^{M-1} \log \sigma_i(\boldsymbol{x}) + \sum_{i=1}^{M} \log z_i, \\ \ell_{rec} &= \sum_{i=1}^{M} \left(x_i - \sum_{j=1}^{N} A_{ij} z_j \right)^2, \\ C &= \frac{1}{2} \log(2\pi) + \log \Gamma(M) - \frac{N}{2} \log(2\pi\sigma^2), \end{aligned}$$

representing the point-wise entropy of the variational distribution, the reconstruction loss, and the constant term, respectively.

Chapter 3: State of the Art

3.1 Identifiability in Latent Variable Models

Identifiability is an essential property of statistical models, which guarantees interpretability of estimated parameters and latent variables, independently of estimation method. This property implies that distinct parameter values correspond to distinct distributions over the observations, allowing for the unique recovery of model parameters and latent variables from data. Formally, a parametric model $\mathcal{P} = \{p_\theta : \theta \in \Theta\}$ defined over a set of observations $x \in \mathcal{X}$ is identifiable if the mapping $\theta \mapsto p_\theta(x)$ is injective, in other words:

$$p_\theta(x) = p_{\theta^*}(x) \implies \theta = \theta^*, \quad \forall x \in \mathcal{X}.$$

In practical scenarios, we are often interested in models that are identifiable up to a certain class of transformations.

Definition 1. *[32] Let $\sim$ be an equivalence relation on the parameter space Θ. We say that a model is identifiable up to $\sim$ (or $\sim$-identifiable) on $\mathcal{X}$ if*

$$p_\theta(x) = p_{\theta^*}(x) \implies \theta \sim \theta^*, \quad \forall x \in \mathcal{X}, \tag{3.1}$$

where $p_\theta(x)$ denotes the probability distribution parameterized by θ. The elements of the quotient space $\Theta/\sim$ are referred to as the identifiability classes.

In the following, we will encounter various types of equivalence relations, specific to the generative model under consideration. Linear equivalence relations appear in deterministic nonlinear models, where identifiability is only guaranteed up to a linear transformation of the latent variables. A stricter form of equivalence arises when latent components can be uniquely recovered up to permutation and scaling transformations.

As an example, consider the factor analysis model,

$$\boldsymbol{x} = A\boldsymbol{z} + \boldsymbol{v}, \tag{3.2}$$

where identifiability implies the equivalence relation on the estimated mixing matrix A. The equivalence relation for this model is defined as follows:

Definition 2. *Let $\sim$ be an equivalence relation on $\mathbb{R}^{N \times M}$ defined as follows: $A \sim A^*$ if and only if there exists a matrix $W \in \mathbb{R}^{N \times N}$, such that*

$$A = WA^* \quad \forall \boldsymbol{x} \in \mathcal{X},$$

If W is invertible, we denote this relation by $\sim_L$. If W is a block permutation matrix, we denote it by $\sim_P$, if W is a scaled permutation matrix, we denote it by $\sim_S$.

However, when the model assumes invariance in latent distribution $p(\boldsymbol{z})$, identifiability breaks down. For instance, in the Gaussian factor analysis,

$$\boldsymbol{x} = A\boldsymbol{z} + \boldsymbol{v}, \quad \boldsymbol{v} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}), \quad \boldsymbol{z} \sim \mathcal{N}(0, \boldsymbol{I}), \tag{3.3}$$

the $p(\boldsymbol{z})$ is spherically symmetric, i.e. any rotation of $\boldsymbol{z}$ preserves the distribution. Therefore, an arbitrary orthogonal transformation of $\boldsymbol{z}$ will leave $p(\boldsymbol{z})$ unchanged, which implies $\sim_L$ equivalence relation on A. While $p_A(\boldsymbol{x})$ remains constant, these transformations change $p_A(\boldsymbol{z}|\boldsymbol{x})$, making it impossible to uniquely recover the true posterior distribution and the original latent structure in the model, also known as factor rotation indeterminacy [26]. Similarly, optimizing the marginal likelihood of observed data in VAE does not inherently guarantee that the correct joint distribution over observed and latent variables will be learned. This limitation arises in many VAE implementations, which often result in non-identifiable models [39]. Incorporating structural assumptions, such as conditional or constrained priors, helps break this symmetry and permits unique recovery of the model's parameters and latent variables.

Identifiability typically refers to the ability to uniquely determine the model parameters, but it can be extended to the latent variables, though defining it rigorously can be challenging. In a noise-free factor analysis model, knowing the mixing matrix A allows retrieval of the latent components $\boldsymbol{z}$ via the (pseudo)inverse of A. However, in the presence of noise, full recovery of $\boldsymbol{z}$ from A alone is not possible, instead identifiability refers to the ability to recover the posterior $p_A(\boldsymbol{z}|\boldsymbol{x})$ [33].

In the following, we discuss linear and nonlinear identifiability in key ICA and VAE models, focusing on the role of structural assumptions, and general methodology for establishing identifiability. We provide high-level summaries of the proofs, while emphasizing key techniques used in establishing identifiability guarantees. We begin by discussing identifiability in ICA, a foundational framework for recovering statistically independent, non-Gaussian latent components from observed linear mixtures. ICA's identifiability principles [12], serve as a blueprint, introducing essential techniques and the methodology of identifiability proofs. Next, we discuss the nonlinear ICA (nICA) [29], that leverages auxiliary observations to disentangle latent sources. This model establishes a framework for handling nonlinear identifiability. We further explore latent variable models with simplex-structured latent spaces, which introduce an alternative approach to identifiability based on geometric constraints on the prior. In the Simplex-Constrained Post-Nonlinear Model (SC-PNM), the identifiability is achieved by constraining the latent space to a unit simplex. This model leverages the geometry of the simplex to establish identifiability for a specific class of nonlinear mappings, and introduces techniques that allow to convert simplex constraints into equivalence relations.

Finally, we examine identifiability in noisy generative models. Formally, in the presence of noise we cannot identify latents from the observed data, even if we know the true parameters. In this case, identifiability of the model is framed as matching the true posterior and the conditional likelihood. We first discuss the auxiliary-variable-based VAE (iVAE) that utilizes auxiliary variables to structure the latent space and condition the prior, which enables nonlinear identifiability. This model introduces important techniques for establishing equivalence relations in noisy models. The Probabilistic Simplex Component Analysis (PRISM) further adapts these probabilistic techniques to simplex-structured LVMs, applying the framework to cases with geometrically constrained latent variables. Through techniques such as conditional priors and structured variational inference, iVAE and PRISM establish identifiability in complex probabilistic models where traditional approaches would struggle.

3.2 Deterministic Identifiability

3.2.1 Independent Component Analysis

ICA is closely related to the Gaussian factor model. It assumes that the observed vector $\boldsymbol{x} \in \mathbb{R}^N$ is a linear combination of latent, statistically independent sources $\boldsymbol{z} \in \mathbb{R}^N$, represented as

$$\boldsymbol{x} = A\boldsymbol{z}, \tag{3.4}$$

where $A \in \mathbb{R}^{N \times N}$ is an invertible mixing matrix [12, 31, 45, 25]. Here, $\boldsymbol{z} \in \mathbb{R}^N$ represents independent sources that are non-Gaussian, zero-centered, and with unit variance.

The non-Gaussian assumption fundamentally distinguishes ICA from classical factor analysis, and enables identifiability. It allows exploiting additional information contained in higher-order statistical properties [45, 25], such as kurtosis, to break the rotational symmetry and achieve identifiability. ICA typically assumes that the number of observed variables matches the number of independent sources, i.e., $M = N$, so that A is square and invertible. In practice, the dimension is often reduced by PCA as a preprocessing step. Hence, ICA can be seen as a "factor rotation", considering the principal components as estimates of factors [26]. Te lack of a noise term in ICA is compensated by the high number of components, which together account for all data variance, implicitly capturing both noise and signal.

ICA identifies latent components by finding an invertible transformation that maximizes the non-Gaussianity of the resulting outputs. The intuition behind this approach stems from the Central Limit Theorem, which implies that a sum of independent random variables is generally more Gaussian than each individual variable, especially if they share the same distribution. Therefore, among possible transformations, the one producing the least Gaussian (most non-Gaussian) component aligns with a single source variable, maximizing the non-Gaussianity of each transformed component. For further insights, refer to [25].

However, ICA involves two fundamental ambiguities: first, the ordering of components remains undefined, and second, each component can only be estimated up to arbitrary scale and sign, as multiplying a component by a constant with a corresponding adjustment in the columns of A leaves the data distribution unchanged. Conventionally, setting component variances to unity reduces this scale ambiguity, standardizing the components to a *white* form, though the exact variance of the sources themselves remains unidentifiable.

In terms of the definition in (2), we say that ICA is identifiable up to the equivalence relation $\sim_S$, i.e.

$$p_A(\boldsymbol{x}) = p_{A^*}(\boldsymbol{x}) \implies A = SA^*, \quad \forall \boldsymbol{x} \in \mathbb{R}^N, \tag{3.5}$$

where S is a scaled permutation matrix. In the ICA context, identifiability implies that if two different mixing matrices A and A^* yield the same distribution of $\boldsymbol{x}$, then they must be equivalent up to scaling and permutation of columns. The following theorem provides a formal statement and the high-level proof of identifiability for ICA:

Theorem 1 (ICA Identifiability [25]). *For a linear mixture model $\boldsymbol{x} = A\boldsymbol{z}$, where z_i are independent components, the mixing matrix A is $\sim_S$-identifiable, i.e., it has exactly one non-zero entry in each row and column, under the following conditions:*

Assumption 1.1. *Each independent component z_i has finite variance.*

Assumption 1.2. *The log-pdfs $\log p_{z_i}(z_i)$ are twice continuously differentiable.*

Assumption 1.3. *The components z_i are non-Gaussian.*

Proof. Without loss of generality, A is assumed orthogonal, which can be achieved by whitening $\boldsymbol{x}$ beforehand. Whitening standardizes the covariance of $\boldsymbol{z}$ and forces A to be orthogonal, simplifying the proof.

The identifiability proof in ICA proceeds in three steps: First, $p(\boldsymbol{x})$ is expressed in terms of the densities of the independent, non-Gaussian latent components $\boldsymbol{z}$, leveraging their independence to factorize $p(\boldsymbol{x})$. Next, second derivatives of $\log p(\boldsymbol{x})$ with respect to $\boldsymbol{x}$ are taken, where independence implies that cross-terms vanish, leading to a matrix equation that isolates each z_i's influence. Finally, this matrix equation forms an eigenvalue decomposition (EVD). The orthogonality of A and distinct values from the non-Gaussianity of $\boldsymbol{z}$ ensure a unique EVD up to permutation and sign changes, establishing the identifiability of $\boldsymbol{z}$.

We can find the probability density function of $\boldsymbol{x}$ by transforming the density of $\boldsymbol{z}$:

$$p(\boldsymbol{x}) = \prod_{i=1}^{n} p_{z_i}(y_i) \left|\det(A^\top)\right|, \tag{3.6}$$

where

$$y_i = \sum_{j=1}^{n} a_{ji} x_j$$

and p_{z_i} denotes the pdf of each component z_i. Taking the logarithm of $p(\boldsymbol{x})$, we obtain

$$\log p(\boldsymbol{x}) = \sum_{i=1}^{n} \log p_{z_i}(y_i), \tag{3.7}$$

where $\log \left|\det(A^\top)\right| = 0$ due to the orthogonality of A.

Since the observed variables x_i are independent, we can decompose $\log p(\boldsymbol{x}) = \sum_{i=1}^{n} f_i(x_i)$ for some functions f_i, indicating that all second-order cross-derivatives must be zero:

$$\frac{\partial^2 \log p(\boldsymbol{x})}{\partial x_k \partial x_l} = \sum_{i=1}^{n} a_{ki} a_{li} \left(\log p_{z_i}(y_i)\right)'' = 0, \quad \forall k \neq l. \tag{3.8}$$

In matrix form, this system can be expressed as the eigenvalue decomposition (EVD):

$$A^\top \operatorname{diag}\left(\log p_{z_i}(y_i)\right)'' A = \operatorname{diag}(c_i(\boldsymbol{x}; A)) \tag{3.9}$$

where $c_i(\boldsymbol{x}; A)$ are unknown scalar functions, and which is valid for all $\boldsymbol{x}$.

The uniqueness of EVD here shows why Gaussian components lack identifiability. For Gaussian densities, $\log p_{z_i}$ is quadratic, yielding a constant second derivative $(\log p_{z_i})''$ and reducing the left side of equation (3.9) to an identity matrix, permitting any orthogonal transformation of A. In contrast, for non-Gaussian densities, the second derivative of $\log p_{z_i}$ is non-constant, allowing to choose points $\boldsymbol{x}$ such that each diagonal entry is distinct. By the uniqueness of EVD, this requirement enforces that the components of A are identifiable up to permutation and scaling of columns, concluding the proof. □

An alternative approach to the proof utilizes the Fourier domain by working with characteristic functions $\hat{p}$. Here, $p(\boldsymbol{x})$ and $p_{z_i}(z_i)$ are replaced by their characteristic functions, eliminating the need for the Jacobian in the initial equations. This approach replaces the smoothness assumption of the probability density functions with the requirement of continuous second derivatives for the characteristic functions $\hat{p}_{z_i}$, which is slightly more restrictive than assuming finite variances.

3.2.2 Nonlinear ICA with Auxiliary Variables

The fundamental concept of ICA is extended here to the nonlinear setting, where the observed vector $\boldsymbol{x} \in \mathbb{R}^N$ is generated by a nonlinear, invertible mixing function $\boldsymbol{f} : \mathbb{R}^N \to \mathbb{R}^N$:

$$\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{z}). \tag{3.10}$$

This mixing function $\boldsymbol{f}$ is assumed to be smooth, with continuous second derivatives, and it need not follow any specific functional form. Consequently, $\boldsymbol{f}$ can be modeled by a neural network, and empirical results suggest that it generally maintains invertibility without explicit constraints [29].

To estimate the model, a contrastive learning approach is employed, which distinguishes between real and randomized datasets. Specifically, two datasets are defined as $\tilde{\boldsymbol{x}} = (\boldsymbol{x}, \boldsymbol{u})$ and $\tilde{\boldsymbol{x}}^* = (\boldsymbol{x}, \boldsymbol{u}^*)$, where $\boldsymbol{u}^*$ is a randomly selected, independent sample drawn from the distribution of $\boldsymbol{u}$, generated by permuting the empirical observations of $\boldsymbol{u}$. The estimation procedure uses a nonlinear logistic regression model (e.g., a neural network) with a regression function defined as

$$r(\boldsymbol{x}, \boldsymbol{u}) = \sum_{n=1}^{N} \psi_n(h_n(\boldsymbol{x}), \boldsymbol{u}), \tag{3.11}$$

which yields the posterior probability of the original class as $(1 + \exp(-r(\boldsymbol{x}, \boldsymbol{u})))^{-1}$. This contrastive approach effectively leverages the auxiliary variable $\boldsymbol{u}$ for enhanced separation of the latent structure.

For identifiability, the approach assumes that each latent component z_i in the vector $\boldsymbol{z} = [z_1, \ldots, z_N]^\top$ is conditionally dependent on an observed m-dimensional auxiliary variable $\boldsymbol{u}$, while being conditionally independent of the other components:

$$\log p(\boldsymbol{z}|\boldsymbol{u}) = \sum_{n=1}^{N} q_n(z_n, \boldsymbol{u}), \tag{3.12}$$

for some smooth functions q_i. The auxiliary variable $\boldsymbol{u}$ is application-specific and may represent temporal information (e.g., previous time steps in time series), spatial indices (e.g., pixel indices for image data), or other contextual variables such as class labels. This formulation generalizes the independence assumption in linear ICA by introducing conditional dependencies modulated by $\boldsymbol{u}$, which is essential for separating the components in nonlinear settings.

Theorem 2 (nICA Identifiability[29])**.** *Assume the observed data $\boldsymbol{x}$ follows the nonlinear ICA model with auxiliary variables in* (3.10)*, and is optimized using regression function* (3.11)*. As the amount of data $T \to \infty$, the functions $h_i(\boldsymbol{x})$ in the learned regression function recover the independent components z_i, up to invertible scalar transformations, if the following assumptions hold:*

Assumption 2.1. *The nonlinear mixing function $\boldsymbol{f} : \mathbb{R}^N \to \mathbb{R}^N$ is invertible and smooth.*

Assumption 2.2. *The latent components z_i are conditionally dependent on an auxiliary variable $\boldsymbol{u}$, with conditional independence among the components given $\boldsymbol{u}$ in* (3.12)*.*

Assumption 2.3. *For any $\boldsymbol{y} \in \mathbb{R}^N$, there exist $2N+1$ values of $\boldsymbol{u}$, denoted by $\boldsymbol{u}_j$, $j = 0, \ldots, 2N$, such that the set of vectors*

$$\{w(\boldsymbol{y}, \boldsymbol{u}_j) - w(y, \boldsymbol{u}_0) \mid j = 1, \ldots, 2N\}$$

is linearly independent, where

$$w(\boldsymbol{y}, \boldsymbol{u}) = \left(\frac{\partial q_1}{\partial z_1}(y_1, \boldsymbol{u}), \ldots, \frac{\partial q_N}{\partial z_N}(y_N, \boldsymbol{u}), \frac{\partial^2 q_1}{\partial z_1^2}(y_1, \boldsymbol{u}), \ldots, \frac{\partial^2 q_N}{\partial z_N^2}(y_N, \boldsymbol{u}) \right).$$

Assumption 2.4. *The regression function in the learning model has universal approximation capability and is trained to discriminate between pairs $(\boldsymbol{x}, \boldsymbol{u})$ and $(\boldsymbol{x}, \boldsymbol{u}^*)$ with $\boldsymbol{u}^*$ permuted independently of $\boldsymbol{x}$.*

This approach builds on the ICA proof's foundation and leverages variability in auxiliary variables to resolve the linear system formed by cross-derivative equations.

Proof. According to [23], after convergence with infinite data, the learned regression function approximates the difference in log-densities between two classes. Specifically, the regression function can be expressed as:

$$\sum_{i=1}^{N} \psi_i(h_i(\boldsymbol{x}), \boldsymbol{u}) = \sum_{i=1}^{N} q_i(g_i(\boldsymbol{x}), \boldsymbol{u}) + \log p(\boldsymbol{u}) - \log p_z(\boldsymbol{v}(\boldsymbol{x}))$$

where $\boldsymbol{g} = \boldsymbol{f}^{-1}$, p_z is the marginal density over the latent components, and $\boldsymbol{h}(\boldsymbol{x})$ and $\boldsymbol{v}(\boldsymbol{y}) = \boldsymbol{g}(\boldsymbol{h}^{-1}(\boldsymbol{y}))$ are derived from a change of variables. In this equation, the Jacobian determinants and the marginal $\log p(\boldsymbol{u})$ terms cancel. This expression parallels the ICA framework, where the density of $\boldsymbol{x}$ is expressed as a product of the densities of independent components. However, here the inclusion of auxiliary variables $\boldsymbol{u}$ introduces additional structure, which will later facilitate identifiability.

The second step involves taking first and second derivatives of the transformed density expression to analyze the dependencies between components. Differentiating both sides of the equation with respect to y_j yields:

$$\psi_j'(y_j, \boldsymbol{u}) = \sum_{n=1}^{N} q_n'(v_i(\boldsymbol{y}), \boldsymbol{u}) v_j^n(\boldsymbol{y}) - \bar{q}_j(\boldsymbol{y})$$

where $v_j^n(\boldsymbol{y}) = \frac{\partial v_n}{\partial y_j}$ and $\bar{q}(\boldsymbol{y}) = \log p_z(\boldsymbol{v}(\boldsymbol{y}))$. A subsequent derivative with respect to $y_{j'}$ (for $j' \neq j$) introduces cross-derivative terms:

$$\sum_{n=1}^{N} q_n''(v_i(\boldsymbol{y}), \boldsymbol{u}) v_j^n(\boldsymbol{y}) v_{j'}^n(\boldsymbol{y}) + q_n'(v_n(\boldsymbol{y}), \boldsymbol{u}) v_{jj'}^n(\boldsymbol{y}) - \bar{q}_{jj'}(\boldsymbol{y}) = 0$$

where $v_{jj'}^n(\boldsymbol{y})$ represents second-order cross-derivatives. This step mirrors the cross-derivative analysis in the linear ICA proof, where the independence of components implies vanishing cross-terms. However, here, the introduction of auxiliary variables and nonlinearity results in new terms, requiring a more complex analysis to establish that each component v_n depends on only one y_n.

The final step utilizes the variability in auxiliary variables $\boldsymbol{u}$ to resolve the linear system formed by these cross-term equations. By collecting the cross-derivative conditions into a matrix $M(\boldsymbol{y})$ with size $N(N-1)/2 \times 2N$, the equations can be expressed as:

$$M(\boldsymbol{y})\boldsymbol{w}(\boldsymbol{y}, \boldsymbol{u}) = \boldsymbol{c}(\boldsymbol{y})$$

with $\boldsymbol{w}(\boldsymbol{y}, \boldsymbol{u})$ representing terms involving the first and second derivatives of q_n. Auxiliary variables $\boldsymbol{u}$ are selected with values $\boldsymbol{u}_0, \boldsymbol{u}_1, \ldots, \boldsymbol{u}_{2N}$, yielding:

$$M(\boldsymbol{y})(\boldsymbol{w}(\boldsymbol{y}, \boldsymbol{u}_1) - w(y, \boldsymbol{u}_0), \ldots, \boldsymbol{w}(\boldsymbol{y}, \boldsymbol{u}_{2N}) - \boldsymbol{w}(\boldsymbol{y}, \boldsymbol{u}_0)) = 0$$

The linear independence assumption on the columns of w ensures that $M(\boldsymbol{y})$ must be zero, forcing $a_n(\boldsymbol{y}) = 0$ and $b_n(\boldsymbol{y}) = 0$ for each n. This confirms that each row of the Jacobian of v has only one non-zero entry, meaning each v_n is exclusively a function of a single y_n.

To conclude, the continuity and invertibility of the Jacobian imply that each v_n consistently depends on one y_n, as any deviation would introduce singularities. Thus, each v_n is an invertible function of a single y_n, completing the identifiability proof.

□

3.2.3 Simplex Constrained Post-Nonlinear Mixture

Another way to break the symmetry of the latent space is by constraining the latent variables domain to a lower-dimensional manifold. The Simplex-Constrained Post-Nonlinear Mixture (SC-PNM) model is a nonlinear generalization of the linear latent component model, where the identifiability is enabled by constraining the latent space to the unit simplex [55, 41]. This model applies to scenarios where the data generation process can be decomposed into a linear mixing stage followed by unknown nonlinear scalar distortions on each observation channel. The model can be expressed as follows:

$$\boldsymbol{x} = \boldsymbol{f}(A\boldsymbol{z}), \tag{3.13}$$

where $\boldsymbol{f}(\boldsymbol{y}) = [f_1(y_1), \ldots, f_N(y_N)]^\top$ is a component-wise nonlinear continuous function, and $\boldsymbol{z} \in \Delta^{M-1}$.

The SC-PNM problem is formulated as constructing a transformation $\boldsymbol{g} : \mathbb{R}^N \to \mathbb{R}^N$ that inversely identifies the nonlinear functions $\boldsymbol{f}(\cdot)$, satisfying the following conditions:

$$\sum_{n=1}^{N} g_n(x_n) = 1, \quad \forall \boldsymbol{x} \in \mathcal{X}, \tag{3.14}$$

where $\mathcal{X} = \{\boldsymbol{x} \in \mathbb{R}^N | \boldsymbol{x} = \boldsymbol{f}(A\boldsymbol{z}), \forall \boldsymbol{z} \in \Delta^{M-1}\}$, and each $g_n : \mathbb{R} \to \mathbb{R}$ is an invertible, scalar-valued function.

To solve for the nonlinear functions $\boldsymbol{g}(\cdot)$, SC-PNM model leverages a neural network-based autoencoder framework. The objective is to reconstruct each observation $\boldsymbol{x}_t$ through an invertible transformation and a subsequent nonlinear mapping. This can be formalized as the following optimization problem:

$$\min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{t=1}^{N} \|\boldsymbol{q}(\boldsymbol{g}(\boldsymbol{x}_t)) - \boldsymbol{x}_t\|_2^2, \tag{3.15}$$

where $\boldsymbol{q}(\cdot) = [q_1(\cdot), \ldots, q_N(\cdot)]^\top$ is a neural decoder. The reconstruction error is minimized under the constraint:

$$\mathbf{1}^\top \boldsymbol{g}(\boldsymbol{x}_t) - 1 = 0, \quad t = 1, \ldots, T \tag{3.16}$$

where this condition ensures that the transformed data $\boldsymbol{g}(\boldsymbol{x}_t)$ remains on the simplex.

Before proceeding to the identifiability theorem we provide two technical lemmas, that will be used in our formalism in Chapter 3.

Fact 1. *Assume $M \geq 3$, and consider $\boldsymbol{z} \in \overline{\Delta}^{M-1}$. Then, $\partial z_i / \partial z_j = 0$ for $i \neq j$ where $i, j \in [M-1]$.*

Proof. For $\boldsymbol{z} \in \overline{\Delta}^{M-1}$, we have $M - 1$ free variables, e.g., z_i for $i = 1, \ldots, M-1$. Assume $i, j \in [M-1]$ and $i \neq j$. For any fixed z_i, z_j can take any values within a nonempty continuous domain. Thus, treating z_i as a function of z_j implies $\partial z_i / \partial z_j = 0$ for $\boldsymbol{z} \in \overline{\Delta}^{M}$. □

Theorem 3 (SC-PNM Identifiability [41])**.** *Almost surely, any solution $\boldsymbol{g} = [g_1, \ldots, g_N]^\top$, that satisfies* (3.14) *and the following assumptions, ensures that each $h_n(y) = g_n \circ f_n(y)$ is affine,* *specifically:*

$$h_n(y) = c_n y + d_n, \quad c_n \neq 0, \quad d_n \in \mathbb{R}. \tag{3.17}$$

Moreover, if $\sum_{n=1}^{N} d_n \neq 1$, *we have* $\boldsymbol{h}(A\boldsymbol{z}) = \hat{A}\boldsymbol{z}$, *where* $\hat{A} = DA$ *with a full-rank diagonal matrix* D, *i.e.* $\boldsymbol{h}$ *is a linear function on a simplex* $\mathcal{A} = \{A\boldsymbol{z} \in \mathbb{R}^M \mid \mathbf{1}^\top \boldsymbol{z} = 1, \boldsymbol{z} \geq 0\}$, *or the model is* $\sim_S$*-identifiable on* $\mathcal{A}$.

Assumption 3.1. *The mixing matrix* $A \in \mathbb{R}^{N \times M}$ *is drawn from a continuous distribution.*

Assumption 3.2. *Each* $h_n = g_n \circ f_n$ *is twice differentiable and invertible.*

Assumption 3.3. *The dimension constraints* $3 \leq M \leq N \leq \frac{M(M-1)}{2}$ *hold.*

Proof. Solving criterion (3.14) leads to

$$\sum_{i=1}^{N} \hat{h}_i \left(a_{i,1} z_1 + \cdots + a_{i,M} \left(1 - \sum_{j=1}^{M-1} z_j \right) \right) = 1,$$

using $z_M = 1 - \sum_{j=1}^{M-1} z_j$, and Fact 1 ensures that $\partial z_i / \partial z_j = 0$ for $i \neq j$. Taking second-order derivatives with respect to z_i and z_j for $i, j \in [M-1]$ results in the system:

$$G\hat{\boldsymbol{h}}'' = \begin{bmatrix} (b_1 \odot b_1)^\top \\ \vdots \\ (b_{M-1} \odot b_{M-1})^\top \\ (b_1 \odot b_2)^\top \\ \vdots \\ (b_{M-2} \odot b_{M-1})^\top \end{bmatrix} \begin{bmatrix} \hat{h}_1'' \\ \vdots \\ \hat{h}_N'' \end{bmatrix} = 0, \tag{3.18}$$

where $\boldsymbol{b}_i = [a_{1,i} - a_{1,M}, \ldots, a_{N,i} - a_{N,M}]^\top$ for $i = 1, \ldots, M-1$, and $B = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{M-1}]$. Here, G has dimensions $\frac{M(M-1)}{2} \times N$, and we aim to establish that $\operatorname{rank}(G) = N$.

This rank condition can be shown by constructing a specific case where an $N \times N$ submatrix of G has full rank. Consider a scenario in which B is a Vandermonde matrix, such that $b_i = [1, z_i, z_i^2, \ldots, z_i^{N-1}]^\top$ with $z_i \neq z_j$. Such a matrix B can be constructed by ensuring that the first $N-1$ rows of $A^\top$ form a Vandermonde matrix, with the last row containing all zeros. By selecting $\tilde{M}$ columns from this Vandermonde matrix B, with $\tilde{M} \leq M-1$, we satisfy the inequality $N \leq \frac{\tilde{M}(\tilde{M}+1)}{2}$. For simplicity, we assume $N = \frac{\tilde{M}(\tilde{M}+1)}{2}$ in what follows. Now, consider the structure of the submatrix in G formed by the selected rows. This submatrix takes the form:

$$\begin{bmatrix} 1 & z_1^2 & \cdots & z_1^{2(N-1)} \\ \vdots & \vdots & & \vdots \\ 1 & z_{\tilde{M}}^2 & \cdots & z_{\tilde{M}}^{2(N-1)} \\ 1 & z_1 z_2 & \cdots & (z_1 z_2)^{N-1} \\ \vdots & \vdots & & \vdots \\ 1 & z_{\tilde{M}-1} z_{\tilde{M}} & \cdots & (z_{\tilde{M}-1} z_{\tilde{M}})^{N-1} \end{bmatrix}.$$

If we can find a particular case where an $N \times N$ submatrix of G has a non-zero determinant, then by continuity, this property holds almost everywhere for A. One can construct a sequence of values for z_i, such as $z_1 = 1$, $z_2 = 1.1$, $z_3 = 1.11$, and so forth, ensuring that the resulting matrix has full rank. Since the determinant of any $N \times N$ submatrix of G is a polynomial in the entries of A, it follows that this polynomial is either identically zero or non-zero almost everywhere [11] Thus, G has full column rank N almost surely.

Next, we follow the linearity arguments presented in [55], Remark 1, to conclude that $\hat{\boldsymbol{h}}(A\boldsymbol{z})$ is linear in $A\boldsymbol{z}$ if $\sum_{n=1}^{N} d_n \neq 1$. Define

$$T_h(X) = \Big[\boldsymbol{h}(\boldsymbol{x}_1), \quad \boldsymbol{h}(\boldsymbol{x}_2), \quad \cdots, \quad \boldsymbol{h}(\boldsymbol{x}_T)\Big],$$

where $X = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_T] \in \mathbb{R}^{M \times T}$. Given that $\boldsymbol{h}$ is affine, we can write

$$T_h(X) = DX + \boldsymbol{c}\mathbf{1}_T^\top,$$

where $D = \operatorname{diag}(d_1, \ldots, d_M)$ is a diagonal matrix and $\boldsymbol{c} = [c_1, c_2, \ldots, c_M]^\top$. To establish that T_h is linear, consider the row sum of $T_h(X)$:

$$\mathbf{1}_M^\top T_h(X) = \mathbf{1}_M^\top DX + \mathbf{1}_M^\top \boldsymbol{c}\mathbf{1}_T^\top.$$

Condition in 3.16 yields

$$\mathbf{1}_M^\top T_h(X) = \mathbf{1}_T^\top,$$

which implies

$$\mathbf{1}_T^\top = \frac{\mathbf{1}_M^\top DX}{1 - \mathbf{1}_M^\top \boldsymbol{c}}.$$

Consequently, $T_h(X)$ is linear, and we can write

$$T_h(X) = \left(\mathbf{I} + \frac{\boldsymbol{c}\mathbf{1}_M^\top}{1 - \mathbf{1}_M^\top \boldsymbol{c}}\right) DX,$$

thus proving that $\boldsymbol{h}$ is linear, given that $\sum_{n=1}^{N} c_n \neq 1$ (this is assumed to be true given that □

Theorem 3 establishes that identifiability in SC-PNM models is guaranteed only if $N \leq \frac{M(M-1)}{2}$. This condition is counterintuitive from a conventional LMM perspective, where adding more channels N typically increases flexibility and improves performance. However, in SC-PNM learning, having more channels can complicate the model, as each additional $g_n(\cdot)$ introduces unknown nonlinearity that must align with the overall solution structure in (3.18). In cases where N exceeds this threshold, SC-PNM's identifiability cannot be guaranteed with criterion 3.14, as shown in Theorem 4:

Theorem 4. *If $M > \frac{K(K-1)}{2}$, then solutions f_m satisfying criterion* (3.14) *can lead to $h_m = g_m \circ f_m$ functions that are not affine.*

This finding highlights that criterion 3.14 alone is insufficient when M is large relative to K. Fortunately, a modification to the criterion resolves this: instead of applying it globally, one can impose the constraint segment-by-segment as follows:

$$\mathbf{1}^\top \boldsymbol{f}\left([\boldsymbol{x}]_{(p-1)K+1:pK}\right) = 1, \quad p \in \left[\frac{M}{K}\right],$$

assuming M/K is an integer. For non-integer cases, overlapping segments provide a practical extension, ensuring the SC-PNM model remains identifiable and scalable for higher-dimensional data.

Finding a feasible solution for Problem (3.15) is challenging, as feasibility requires satisfying all equality constraints. Since any Karush–Kuhn–Tucker (KKT) point is feasible, efficient KKT-point searching algorithms from nonlinear programming can be leveraged. SC-PNM we utilizes the augmented Lagrangian $L(\boldsymbol{\theta}, \boldsymbol{\lambda})$, defined by:

$$L(\boldsymbol{\theta}, \boldsymbol{\lambda}) = \frac{1}{T}\sum_{t=1}^{T} J_{\boldsymbol{\theta}}(\boldsymbol{x}^{(t)}) + \frac{1}{T}\sum_{t=1}^{T} \lambda_t C_{\boldsymbol{\theta}}(\boldsymbol{x}^{(t)}) + \frac{\rho}{2T}\sum_{t=1}^{T} |C_{\boldsymbol{\theta}}(\boldsymbol{x}^{(t)})|^2,$$

where $J_{\boldsymbol{\theta}}(\boldsymbol{x}^{(t)}) = \|q(\boldsymbol{f}(\boldsymbol{x}^{(t)})) - \boldsymbol{x}^{(t)}\|^2$ and $C_{\boldsymbol{\theta}}(\boldsymbol{x}^{(t)}) = \mathbf{1}^\top \boldsymbol{f}(\boldsymbol{x}^{(t)}) - 1$. The dual variables $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_T]$ and $\rho > 0$ control constraint adherence.

The algorithm iteratively updates $\boldsymbol{\theta}$ and $\boldsymbol{\lambda}$ as follows:

$$\boldsymbol{\theta}^{i+1} \leftarrow \arg\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta}, \boldsymbol{\lambda}^i) \quad \text{(inexact minimization)},$$

$$\lambda_t^{i+1} \leftarrow \lambda_t^i + \rho^i C_{\boldsymbol{\theta}^{i+1}}(\boldsymbol{x}^{(t)}), \quad \rho^{i+1} \leftarrow \kappa\rho^i,$$

where $\kappa > 1$ is a fixed parameter. This method belongs to augmented Lagrangian techniques [4]. Classic results ensure that if $\|\nabla L(\boldsymbol{\theta}^{i+1}, \boldsymbol{\lambda}^i)\|_2^2 \leq \epsilon_i \to 0$, then all limit points are KKT points.

In practice, SGD-based optimizers like Adam are suitable for updating neural network parameters $\boldsymbol{f}$ and q. As $\rho_i \to \infty$ may cause instability, practical alternatives include incrementing ρ_i gradually or keeping it fixed, which both perform well in simulations. Nevertheless, the algorithm exhibits slow convergence, particularly for large-scale problems as it requires multiple passes over the data to ensure constraint satisfaction.

Although the SC-PNM model provably removes the post-nonlinear distortions, it retains the linear ambiguity, and does not support direct identification of mixing matrix and latent components. To address this, the SC-PNM model can be combined with linear SCA methods to facilitate the unmixing and identification of latent components. Commonly used methods include Simplex-Structured Matrix Factorization (SSMF) [20, 19, 21], and the Minimum-Volume Enclosing Simplex (MVES) [20, 19, 53] algorithms.

3.3 Probabilistic Identifiability

3.3.1 Variational Autoencoders with Conditional Prior

In probabilistic terms, identifiability implies that any two different parameter choices, $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$, yielding the same marginal density $p_{\boldsymbol{\theta}}(\boldsymbol{x})$, must have identical joint distributions $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})$. This equivalence means that if we find a parameter $\boldsymbol{\theta}$ such that $p_{\boldsymbol{\theta}}(\boldsymbol{x}) = p_{\boldsymbol{\theta}^*}(\boldsymbol{x})$, then $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) = p_{\boldsymbol{\theta}^*}(\boldsymbol{x}, \boldsymbol{z})$, indicating we have recovered the correct prior, $p_{\boldsymbol{\theta}}(\boldsymbol{z}) = p_{\boldsymbol{\theta}^*}(\boldsymbol{z})$, and posterior, $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}) = p_{\boldsymbol{\theta}^*}(\boldsymbol{z}|\boldsymbol{x})$. For VAEs, this identifiability allows using the posterior $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ to infer the latent sources $\boldsymbol{z}^*$ from the data $\boldsymbol{x}$.

Achieving identifiability in VAEs thus requires modifying the latent structure to overcome the indeterminacies that arise with unconditional latent priors. For example, using a latent prior that is conditioned on additional observations, such as labels or time indices, can establish identifiability by introducing a structured latent space [32]. Under these conditions, if the learned marginal distribution $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ matches the true data distribution, then the learned joint distribution $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})$ will also match the true joint distribution. This model is closely related to the nICA formalism discussed above, but framed as a probabilistic generative model. It allows for guaranteed removal of nonlinear mixing, up to component-wise transformations.

The proposed model assumes a noisy nonlinear model:

$$\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{z}) + \boldsymbol{v}, \tag{3.19}$$

with an independent noise variable $\boldsymbol{v}$, distributed according to $p_{\boldsymbol{v}}(\boldsymbol{v})$, and a conditionally factorized prior distribution over the latent variables $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{u})$, where $\boldsymbol{u}$ is an additional observed variable. This auxiliary variable $\boldsymbol{u}$ could represent, for example, a time index in a time series, a noisy class label, or a concurrent observation. Formally, given observed variables $\boldsymbol{x} \in \mathbb{R}^N$ and $\boldsymbol{u} \in \mathbb{R}^N$, a latent variable $\boldsymbol{z} \in \mathbb{R}^M$ (with $M \leq N$), and model parameters $\boldsymbol{\theta} = (\boldsymbol{f}, \boldsymbol{T}, \boldsymbol{\lambda})$, the conditional generative model is defined by

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}|\boldsymbol{u}) = p_{\boldsymbol{f}}(\boldsymbol{x}|\boldsymbol{z}) p_{\boldsymbol{T},\boldsymbol{\lambda}}(\boldsymbol{z}|\boldsymbol{u}), \tag{3.20}$$

where

$$p_{\boldsymbol{f}}(\boldsymbol{x}|\boldsymbol{z}) = \phi_{\boldsymbol{v}}(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{z})). \tag{3.21}$$

The function $\boldsymbol{f} : \mathbb{R}^M \to \mathbb{R}^N$ is assumed injective and may be arbitrarily complex, often implemented as a neural network in practice. The latent variable prior $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{u})$ exhibits a conditionally factorial structure such that each component z_i follows a univariate exponential family distribution conditioned on $\boldsymbol{u}$:

$$p_{T,\lambda}(\boldsymbol{z}|\boldsymbol{u}) = \prod_{m}^{M} Q_m(z_m) Z_m(\boldsymbol{u}) \exp\left(\sum_{j=1}^{J} T_{m,j}(z_m) \lambda_{m,j}(\boldsymbol{u})\right), \tag{3.22}$$

where Q_m is the base measure, $Z_m(\boldsymbol{u})$ the normalizing constant, $\boldsymbol{T}_m = (T_{m,1}, \ldots, T_{m,J})$ the sufficient statistics, and $\lambda_m(\boldsymbol{u}) = (\lambda_{m,1}(\boldsymbol{u}), \ldots, \lambda_{m,J}(\boldsymbol{u}))$ the parameters dependent on $\boldsymbol{u}$ through an arbitrary function, e.g., a neural network.

Let $Z \subset \mathbb{R}^M$, $X \subset \mathbb{R}^N$, and $U \subset \mathbb{R}^L$ represent the domain, image of $\boldsymbol{f}$, and the support of $\boldsymbol{u}$, respectively. We denote by $T(\boldsymbol{z}) := (\boldsymbol{T}_1(\boldsymbol{z}_1), \ldots, \boldsymbol{T}_M(\boldsymbol{z}_N))$ and $\lambda(\boldsymbol{u}) := (\boldsymbol{\lambda}_1(\boldsymbol{u}), \ldots, \boldsymbol{\lambda}_m(\boldsymbol{u}))$.

Theorem 5 (iVAE Identifiability [32])**.** *Assume that data is generated from the model defined in Eqs.* (3.21)-(3.22) *with parameters* $\boldsymbol{\theta} = (\boldsymbol{f}, \boldsymbol{T}, \boldsymbol{\lambda})$. *The parameters* $\boldsymbol{\theta}$ *are identifiable up to an invertible linear transformation under the following conditions:*

Assumption 5.1. *The characteristic function* $\phi_{\boldsymbol{v}}$ *of the noise distribution* $p_{\boldsymbol{v}}$ *is non-zero almost everywhere.*

Assumption 5.2. *The function* $\boldsymbol{f}$ *is injective.*

Assumption 5.3. *The sufficient statistics* $T_{i,j}$ *are differentiable almost everywhere and linearly independent on any subset of* Z *of positive measure.*

Assumption 5.4. *There exist* $MJ + 1$ *distinct points* $\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{MJ}$ *such that the matrix* $L = (\lambda(\boldsymbol{u}_1) - \lambda(\boldsymbol{u}_0), \ldots, \lambda(\boldsymbol{u}_{MJ}) - \lambda(\boldsymbol{u}_0))$ *is invertible,*

Theorem 5 establishes a basic identifiability for the generative model in equation (3.20). Assuming data is generated by parameters (f^*, T^*, λ^*), and a learning algorithm provides a consistent estimate of parameters (f, T, λ), the true parameters are $(f^*, T^*, \lambda^*) \sim_L (f, T, \lambda)$. In the absence of noise, this implies that the learned transformation $\tilde{f}$ reconstructs the latent variables z from observations x, up to a linear transformation A and nonlinear mappings T and $\tilde{T}$. With noise, identifiability holds analogously for the posterior distribution of the latents.

The proof proceeds in three steps, establishing identifiability by transforming the noisy observation model to a noiseless equivalent, isolating the sufficient statistics, and proving invertibility.

Proof. We start by assuming two sets of parameters $\boldsymbol{\theta} = (\boldsymbol{f}, \boldsymbol{T}, \boldsymbol{\lambda})$ and $\boldsymbol{\theta}^* = (\boldsymbol{f}^*, \boldsymbol{T}^*, \boldsymbol{\lambda}^*)$ such that:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{u}) = p_{\boldsymbol{\theta}^*}(\boldsymbol{x}|\boldsymbol{u}), \quad \forall(\boldsymbol{x}, \boldsymbol{u}). \tag{3.23}$$

Expanding this equality with the noise model from Eq. (3.21), we have

$$\int p_{T,\lambda}(\boldsymbol{z}|\boldsymbol{u})\phi_{\boldsymbol{v}}(\boldsymbol{x} - f(\boldsymbol{z}))d\boldsymbol{z} = \int p_{\boldsymbol{T}^*,\lambda^*}(\boldsymbol{z}|\boldsymbol{u})\phi_{\boldsymbol{v}}(\boldsymbol{x} - f^*(\boldsymbol{z}))d\boldsymbol{z}. \tag{3.24}$$

Applying a change of variable $\bar{\boldsymbol{x}} = f(\boldsymbol{z})$ and using the Fourier transform $\mathcal{F}$ to both sides, we obtain

$$\mathcal{F}[p_{T,\lambda,f,\boldsymbol{u}}](\omega)\hat{\phi}_{\boldsymbol{v}}(\omega) = \mathcal{F}[p_{\boldsymbol{T}^*,\lambda^*,f^*,\boldsymbol{u}}](\omega)\hat{\phi}_{\boldsymbol{v}}(\omega), \tag{3.25}$$

where $\hat{\phi}_{\boldsymbol{v}}$ is the characteristic function of $\phi_{\boldsymbol{v}}$. Since $\phi_{\boldsymbol{v}}(\omega) \neq 0$ almost everywhere, we conclude

$$p_{T,\lambda,f,\boldsymbol{u}} = p_{\boldsymbol{T}^*,\lambda^*,f^*,\boldsymbol{u}}, \tag{3.26}$$

which implies equality in the noise-free probability densities.

Taking the logarithm of both sides, we substitute $p_{\boldsymbol{T},\boldsymbol{\lambda}}(\boldsymbol{z}|\boldsymbol{u})$ from Eq. (3.22):

$$\begin{aligned}
&\log|\det J_{f^{-1}}(\boldsymbol{x})| + \sum_{n=1}^{N}\left(\log Q_n(f_n^{-1}(\boldsymbol{x})) - \log Z_n(\boldsymbol{u}) + \sum_{j=1}^{J} T_{n,j}(f_n^{-1}(\boldsymbol{x}))\lambda_{n,j}(\boldsymbol{u})\right) \\
&= \log|\det J_{f^{*-1}}(\boldsymbol{x})| + \sum_{n=1}^{N}\left(\log {Q^*}_n({f^*}_n^{-1}(\boldsymbol{x})) - \log {Z^*}_n(\boldsymbol{u}) + \sum_{j=1}^{J} \boldsymbol{T}^*_{n,j}({f^*}_n^{-1}(\boldsymbol{x}))\lambda^*_{n,j}(\boldsymbol{u})\right).
\end{aligned} \tag{3.27}$$

Subtracting the equation for $\boldsymbol{u}_0$ from the remaining equations for $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{MJ}$, we isolate the sufficient statistics:

$$L^T T(\boldsymbol{f}^{-1}(\boldsymbol{x})) = L^T \boldsymbol{T}^*(\boldsymbol{f}^{*-1}(\boldsymbol{x})) + \boldsymbol{b}, \tag{3.28}$$

where $\boldsymbol{b}$ represents constants independent of $\boldsymbol{x}$.

Finally, we define the Jacobian $J_{\boldsymbol{T}}$ of $\boldsymbol{T}$ and consider points $\bar{\boldsymbol{x}}_1, \ldots, \bar{\boldsymbol{x}}_J$ where the Jacobian matrix $Q = (J_{\boldsymbol{T}}(\bar{\boldsymbol{x}}_1), \ldots, J_{\boldsymbol{T}}(\bar{\boldsymbol{x}}_J))$ is full rank, ensuring linear independence. Differentiating the prior equality over all points $\bar{\boldsymbol{x}}_i$, we find:

$$Q = AQ^*. \tag{3.29}$$

Since Q and Q^* are invertible, it follows that A is also invertible, thereby concluding that $(\boldsymbol{f}, \boldsymbol{T}, \boldsymbol{\lambda}) \sim_L (\boldsymbol{f}^*, \boldsymbol{T}^*, \boldsymbol{\lambda}^*)$. □

The first step of this proof provides a blueprint for transforming the noisy model to an equivalent noiseless model in identifiability proofs for probabilistic models with additive Gaussian noise, and reducing the probabilistic identifiability problem to a deterministic one.

3.3.2 Probabilistic Simplex Component Analysis

Recall that the PRISM concerns with the following model (see Section 2.4.2):,

$$\boldsymbol{x} = A\boldsymbol{z} + \boldsymbol{v}, \quad \boldsymbol{z} \in \Delta^{M-1}, \quad \boldsymbol{v} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\mathbf{I}), \tag{3.30}$$

where $A \in \mathbb{R}^{N \times M}$ is affinely independent mixing matrix. Assuming $T \to \infty$, by the law of large numbers, the log likelihood function L_T as defined in (2.19) converges to

$$L(A) = \int_{\mathbb{R}^N} p_{A^*}(\boldsymbol{x}) \log p_A(\boldsymbol{x})\, d\boldsymbol{x}. \tag{3.31}$$

By the Kullback-Leibler divergence, we have $L(A^*) \geq L(A)$, with equality if and only if

$$p_{A^*}(\boldsymbol{x}) = p_A(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in \mathcal{X}, \tag{3.32}$$

suggesting that A^* is identifiable if there exists no non-trivial choice of A satisfying this equality.

To establish the identifiability guarantees, PRISM combines the probabilistic treatment of the identifiability [32], discussed in the previous section, with the simplex geometry.

Theorem 6 (PRISM Identifiability [54])**.** *Equation* (3.32) *holds if and only if* $A = A^*\Pi$*, where* Π *is a permutation matrix. Consequently,* A *is a solution to the maximization of ML in* (3.31) *if and only if* $A = A^*\Pi$*, given the following assumptions hold:*

Assumption 6.1. *The matrix* A^* *is affinely independent.*

Assumption 6.2. *The latent variables* $\boldsymbol{z}_t$ *are independently and identically distributed (i.i.d.), with each* $\boldsymbol{z}_t$ *uniformly distributed on* Δ^M*.*

Assumption 6.3. *The noise variables* $\boldsymbol{v}_t$ *are i.i.d. and independent of the* $\boldsymbol{z}_t$*, with each* $\boldsymbol{v}_t$ *Gaussian distributed with mean* $\mathbf{0}$ *and covariance* $\sigma^2\mathbf{I}$*, where* $\sigma > 0$*.*

Proof. To demonstrate the identifiability of the model parameter A, we consider the specific case $M = N + 1$ with A being affinely independent. In this scenario, the noise-free components $\boldsymbol{x}_t$ in Equation (7) follow a uniform distribution over the simplex formed by the columns of A. This allows us to express the density function $p_A(\boldsymbol{x})$ as follows:

$$p_A(\boldsymbol{x}) = \frac{1}{\text{vol}(A)} \mathbf{1}_{\text{conv}(A)}(\boldsymbol{x}),$$

where $\mathbf{1}_{\text{conv}(A)}(\boldsymbol{x})$ is the indicator function for the convex hull of A, denoting that $p_A(\boldsymbol{x})$ is non-zero only within this convex hull. Applying this uniform simplex distribution to the observed model yields:

$$p_A(\boldsymbol{y}) = \int_{\mathbb{R}^{N-1}} \phi_\sigma(\boldsymbol{y} - \boldsymbol{x}) p_A(\boldsymbol{x})\, d\boldsymbol{x}.$$

To proceed, we leverage the Fourier transform (FT) for simplifying the convolution structure of this integral. By defining the Fourier transform $\hat{f}(\xi)$ of a function $f : \mathbb{R}^n \to \mathbb{R}$ as

$$\hat{f}(\xi) = \int_{\mathbb{R}^n} f(\boldsymbol{x}) e^{-j2\pi\xi^\top \boldsymbol{x}} d\boldsymbol{x},$$

we obtain the FT representation of $p(\boldsymbol{y}; A)$, which implies:

$$p_A(\boldsymbol{x}) = p_{A^*}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \Rightarrow \hat{\phi}_\sigma(\xi)\hat{p}_A(\xi) = \hat{\phi}_\sigma(\xi)\hat{p}_{A^*}(\xi) \quad \forall \xi.$$

Since $\hat{\phi}_\sigma(\xi)$ is always non-zero for any ξ due to its Gaussian form $\hat{\phi}_\sigma(\xi) = e^{-2\pi^2\|\xi\|^2}$, we can simplify to:

$$\hat{p}_A(\xi) = \hat{p}_{A^*}(\xi) \quad \forall \xi.$$

Taking the inverse FT on both sides yields:

$$p_A(\boldsymbol{x}) = p_{A^*}(\boldsymbol{x}) \quad \forall \boldsymbol{x},$$

which implies that the convex hulls of A and A^* must coincide:

$$\text{conv}(A) = \text{conv}(A^*).$$

Thus, the vertices of the convex hulls, representing the columns of A and A^*, must be identical up to permutation, giving us

$$\{a_1, \dots, a_N\} = \{a_1^*, \dots, a_N^*\}.$$

This completes the intuitive proof.

For completeness, it is important to address a technical detail: when $p_A(\boldsymbol{x})$ is discontinuous, $\hat{p}_A(\xi)$ may not be integrable, and thus, its inverse FT might not exist in a strict sense. This, along with the generalization to arbitrary M, is formally resolved in the proof provided in [54].

□

Chapter 4: Methodology

4.1 Probabilistic Post-Nonlinear Simplex Component Analysis

Limitations of deterministic PNL-SCA methods [41, 55] are mainly due to two factors: inability to recover the latent space without additional linear unmixing methods, and deterioration of performance in noisy conditions. At the same time, the probabilistic framework of PRISM [54] and VASCA [37] enable out-of-the-box identification of the latent components and show promising results in handling noisy data. The main objective of this study is to bridge the gap between the deterministic PNL-SCA methods and the probabilistic models, by introducing a generative framework for PNL-SCA. Drawing on methodologies borrowed from [41, 55, 54, 37], discussed in Chapter 3, we establish the first probabilistic identifiability guarantees for PNL-SCA in noisy settings. Specifically, we demonstrate that it is possible to achieve the simultaneous removal of nonlinear distortions and the identifiability of the latent components, addressing the challenges in PNL-SCA models.

Consider the following generation process for the observed random vector $\boldsymbol{x} \in \mathbb{R}^N$:

$$\boldsymbol{x} = \boldsymbol{f}(A\boldsymbol{z}) + \boldsymbol{v}, \tag{4.1}$$

where $\boldsymbol{z} \sim \mathcal{D}(\mathbf{1})$ is a latent variable sampled from a uniform Dirichlet distribution. Here, the latent variable is first linearly transformed by a full column-rank mixture matrix $A \in \mathbb{R}^{N\times M}$. This linear mixture then undergoes a component-wise nonlinear transformation $\boldsymbol{f}(\boldsymbol{x}) = [f_1(x_1), \ldots, f_N(x_N)]^\top$, where each $f_i : \mathbb{R} \to \mathbb{R}$ independently models nonlinear distortions in the corresponding component, and is subsequently perturbed by additive Gaussian noise $\boldsymbol{v} \sim \mathcal{N}(0, \sigma^2 I)$.

The probabilistic structure of the model described by (4.1) is defined by a factorized joint distribution of the observed variable $\boldsymbol{x}$ and the latent variable $\boldsymbol{z}$:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) = p_{\boldsymbol{f},A}(\boldsymbol{x}|\boldsymbol{z})p_{\mathbf{1}}(\boldsymbol{z}), \tag{4.2}$$

where $\boldsymbol{\theta} = (\boldsymbol{f}, A)$ represents the parameters of the model, including the mixture matrix A and the nonlinear transformation $\boldsymbol{f}$. The prior distribution over the latent variable $\boldsymbol{z}$ is uniform over the $(M-1)$-dimensional simplex and is expressed as:

$$p_{\mathbf{1}}(\boldsymbol{z}) = (M-1)!\, \mathbb{1}_{\Delta^{M-1}}(\boldsymbol{z}), \tag{4.3}$$

where $\mathbb{1}_{\Delta^{M-1}}(\boldsymbol{z})$ ensures that $\boldsymbol{z}$ lies within the simplex, and $(M-1)!$ normalizes the distribution. The conditional distribution of $\boldsymbol{x}$ given $\boldsymbol{z}$ is modeled as:

$$p_{\boldsymbol{f},A}(\boldsymbol{x}|\boldsymbol{z}) = \phi_\sigma(\boldsymbol{x} - \boldsymbol{f}(A\boldsymbol{z})), \tag{4.4}$$

where $\phi_\sigma(\boldsymbol{x})$ is the Gaussian density function:

$$\phi_\sigma(\boldsymbol{x}) = \frac{1}{(\sqrt{2\pi}\sigma)^N} \exp\left(-\frac{\|\boldsymbol{x}\|^2}{2\sigma^2}\right). \tag{4.5}$$

This distribution describes $\boldsymbol{x}$ as being centered around the nonlinear transformation $\boldsymbol{f}(A\boldsymbol{z})$, with isotropic Gaussian noise of standard deviation σ. To obtain the marginal distribution of the observed data $\boldsymbol{x}$, we integrate the joint distribution (4.2) over the unit simplex with respect to the Lebesgue measure defined in (2.2):

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = (M-1)! \int \phi_\sigma(\boldsymbol{x} - \boldsymbol{f}(A\boldsymbol{z}))\, \mathbb{1}_{\Delta^{M-1}}(\boldsymbol{z})\, d\boldsymbol{\mu}(\boldsymbol{z}). \tag{4.6}$$

As discussed in Section 3.3.2, optimizing the parameters $\boldsymbol{\theta}$ to maximize the likelihood objective, provides the estimate of the true marginal distribution of the observed data $\boldsymbol{x}$. This equality between the estimated and true marginal distribution facilitates the identifiability analysis of the model, as discussed in the following section.

4.2 Identifiability Guarantees

4.2.1 Nonlinearity Removal

Our goal is to establish the equivalence for the estimated and true model parameters and distributions, that is guaranteed by the maximum likelihood estimation. To this end, we introduce the following equivalence relation for the model in (4.2):

Definition 3. *Let $\sim$ be an equivalence relation on Θ defined as follows: $(f, A) \sim (f^*, A^*)$ if and only if there exist a matrix $U \in \mathbb{R}^{N\times N}$ such that*

$$A^+\boldsymbol{f}^{-1}(\boldsymbol{x}) = UA^{*+}\boldsymbol{f}^{*-1}(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in \mathcal{X}.$$

For an invertible U, the equivalence relation is denoted by $\sim$, for a block permutation matrix, by $\sim_P$, and for a scaled permutation matrix, by $\sim_S$.

This definition implies that equivalent nonlinear functions $\boldsymbol{f}$ and $\boldsymbol{f}^*$ are related by a linear transformation. We say that $\boldsymbol{f}^{-1}$ removes nonlinearity $\boldsymbol{f}^*$ if $\boldsymbol{f} \sim \boldsymbol{f}^*$.

Building on techniques discussed in the previous chapters, we demonstrate that under mild regularity assumptions, maximizing likelihood derived from (4.6) guarantees that the estimated parameters are $\sim$-equivalent to the true values. This result is formalized in the following nonlinear identifiability theorem, which constitutes the central theoretical contribution of this dissertation.

Theorem 7 (Nonlinear Identifiability)**.** *Assuming that the data is generated according to the model* (4.2) *with true parameters $(\boldsymbol{f}^*, A^*)$, the estimated parameters of the model $p_{\boldsymbol{f},A}(\boldsymbol{x})$ in* (4.6) *are identifiable up to a scaled permutation, i.e. $(\boldsymbol{f}, A) \sim_S (\boldsymbol{f}^*, A^*)$, if the following assumptions hold:*

Assumption 7.1. *Functions $f_1, \dots, f_N$ are twice differentiable, and invertible.*

Assumption 7.2. *The matrix $A \in \mathbb{R}^{N\times M}$ has a full column rank.*

Assumption 7.3. *Dimensions of the problem satisfy* $3 \leq M \leq N \leq M(M-1)/2$.

Assumption 7.4. *The set* $\{\boldsymbol{x} \in \mathcal{X} | \phi_\sigma(\boldsymbol{x}) = 0\}$ *has measure zero, where* ϕ_σ *is defined in* (4.4).

Proof. Along the lines of the identifiability proofs in Section 3.3, we prove the theorem in three steps. Assuming the MLE criterion is maximized, the estimated marginal distribution of the data converges to the true distribution. The first step is to simplify the equality between the true and estimated distributions, by removing the additive Gaussian noise, using Lemma 2. This yields an equality between the supports of the noiseless distributions. We next exploit the geometric structure of the probability simplex to transform it into a functional equation for the model parameters, formalized in Lemma 3. Finally, we exploit properties of the Hessian matrix to transform the functional equation into equivalence relations, as stated in Lemma 4.

Suppose we have two sets of parameters $(\boldsymbol{f}, \boldsymbol{A})$ and $(\boldsymbol{f}^*, A^*)$ such that their likelihoods are equal:

$$p_{\boldsymbol{f},\boldsymbol{A}}(\boldsymbol{x}) = p_{\boldsymbol{f}^*,A^*}(\boldsymbol{x}) \quad \text{for all } \boldsymbol{x} \in \mathcal{X}. \tag{4.7}$$

According to Lemma 2, given that Assumption 7.4 is valid, (4.7) implies equality of the noiseless distributions

$$\tilde{p}_{\boldsymbol{f},\boldsymbol{A}}(\boldsymbol{x}) = \tilde{p}_{\boldsymbol{f}^*,A^*}(\boldsymbol{x}) \quad \text{for all } \boldsymbol{x} \in \mathcal{X}, \tag{4.8}$$

where

$$\tilde{p}_{\boldsymbol{f},\boldsymbol{A}}(\boldsymbol{x}) = p(A^+\boldsymbol{f}^{-1}(\boldsymbol{x}))\,\mathrm{vol}(\boldsymbol{A})^{-1}\,\mathrm{vol}(J_{\boldsymbol{f}^{-1}}(\boldsymbol{x})). \tag{4.9}$$

The noiseless density (4.9) can be viewed as a transformation of the prior distribution induced by the nonlinear transformation $\boldsymbol{f}$ and the linear mixture A.

Next, we introduce a change of variables $\boldsymbol{y} = \boldsymbol{f}^{-1}(\boldsymbol{x})$ and denote $\boldsymbol{h} = \boldsymbol{f}^{-1} \circ \boldsymbol{f}^*$. Equation (4.8) yields

$$\frac{\mathbb{1}_{\overline{\mathrm{conv}}(\boldsymbol{A})}(\boldsymbol{y})}{\mathrm{vol}\ A\,\mathrm{vol}\ \boldsymbol{J}_{\boldsymbol{f}}(\boldsymbol{y})} = \frac{\mathbb{1}_{\overline{\mathrm{conv}}(A^\star)}(\boldsymbol{h}(\boldsymbol{y}))}{\mathrm{vol}\ A^\star\,\mathrm{vol}\ \boldsymbol{J}_{\boldsymbol{f}^\star}(\boldsymbol{h}(\boldsymbol{y}))},$$

where we used the inverse function theorem. Given that the denominator is finite and non-zero, we obtain

$$\mathbb{1}_{\overline{\mathrm{conv}}(\boldsymbol{A})}(\boldsymbol{y}) = \mathbb{1}_{\overline{\mathrm{conv}}(A^\star)}(\boldsymbol{h}(\boldsymbol{y})). \tag{4.10}$$

This equation implies that $\boldsymbol{h}$ transforms the convex hull $\overline{\mathrm{conv}}(\boldsymbol{A})$ into convex $\overline{\mathrm{conv}}(A^\star)$:

$$\{\boldsymbol{y} = A\boldsymbol{z} \mid \boldsymbol{z} \in \mathbb{R}^M_{++},\ \mathbf{1}^\top \boldsymbol{z} = 1\} = \{\boldsymbol{y} = \boldsymbol{h}^{-1}(A^\star \boldsymbol{z}) \mid \boldsymbol{z} \in \mathbb{R}^M_{++},\ \mathbf{1}^\top \boldsymbol{z} = 1\}. \tag{4.11}$$

Using Lemma 3, we can now derive a functional equation for the model parameters:

$$\mathbf{1}^\top A^{\star+}\boldsymbol{h}(A\boldsymbol{z}) = 1, \quad \forall \boldsymbol{z} \in \overline{\Delta}^M. \tag{4.12}$$

Be denoting $\boldsymbol{b} = \mathbf{1}^\top A^{*+}$ we can rewrite this funcitonl equation as

$$\boldsymbol{b}\boldsymbol{h}(\boldsymbol{y}) = 1, \quad \forall \boldsymbol{y} \in \overline{\mathrm{conv}}(A). \tag{4.13}$$

We can probe the linearity of the function $\boldsymbol{h}$ by considering the second derivative of (4.13).

This yields a full-rank system of homogenous linear equations for the Hessian matrix of $\boldsymbol{h}$, which implies that $\boldsymbol{h}$ is a linear transformation. According to Lemma 4, if assumptions 7.1, 7.2, and 7.3 are satisfied, the matrix equation in (4.13) implies that $\boldsymbol{h}$ is a linear function. Given that $\boldsymbol{h}$ is a component-wise transformation, this amounts to $\sim_S$ identifiability, i.e. the post-nonlinear distortions are identified up to rescaling. This completes the proof.

□

Theorem 7 establishes a nonlinear form of identifiability for the generative model defined by equations (4.6). Specifically, consider data generated from an original parameter set $(\boldsymbol{f}^*, A^*)$, and let $(\boldsymbol{f}, \boldsymbol{A})$ represent parameters estimated by a consistent learning algorithm in the population limit. The theorem guarantees that these estimated parameters are equivalent to the true parameters up to a scaled permutation, $(\boldsymbol{f}, \boldsymbol{A}) \sim_S (\boldsymbol{f}^*, A^*)$. This result implies that, in the absence of noise, the learned transformation $\boldsymbol{f}$ would map observations to latent mixtures $\boldsymbol{y} = \boldsymbol{f}^{-1}(\boldsymbol{x})$ that are a linear transformation of the true mixtures $\boldsymbol{y}^* = \boldsymbol{f}^{*-1}(\boldsymbol{x})$. In other words, $\boldsymbol{y} = h(\boldsymbol{y}^*)$, and $\boldsymbol{h} = \boldsymbol{f}^{-1} \circ \boldsymbol{f}^*$ is a linear function.

4.2.2 Latent Variables Identification

Unlike the models in Section 3.2, where the latent variables are related to the observed data by a deterministic transformation, in the probabilistic setting, this relationship is stochastic due to the presence of noise. As a result, the latent variables cannot be uniquely identified from the observed data, even if the model parameters are known. In this case identifiability refers to the unique recovery of the posterior distribution over the latent variables, given the observed data. We this in mind, we define the following equivalence relation:

Definition 4 (Probability Equivalence)**.** *Let $\sim$ be an equivalence relation on Θ defined as follows: $p_\theta \sim p_{\theta^*}$ if only if there exists a matrix U such that*

$$p_\theta(\boldsymbol{x}) = \det U p_{\theta^*}(U^{-1}\boldsymbol{x}), \quad \forall \boldsymbol{x} \in \mathbb{R}^N, \tag{4.14}$$

where $U \in \mathbb{R}^{N \times N}$. For an invertible U, the equivalence relation is denoted by $\sim_L$, for a block permutation matrix, by $\sim_P$, and for a scaled permutation matrix, by $\sim_S$.

Lemma 1 (Means of Equivalent Distributions)**.** *Let $p_\theta \sim_L p_{\theta^*}$ be two equivalent distributions, then the means of the distributions are related by*

$$\mathbb{E}_{p_\theta}[\boldsymbol{x}] = U\mathbb{E}_{p_{\theta^*}}[\boldsymbol{x}].$$

Proof. The proof is trivial. Given the equivalence relation (4.14), we have

$$\begin{aligned}\mathbb{E}_{p_\theta}[\boldsymbol{x}] &= \int \boldsymbol{x} p_\theta(\boldsymbol{x}) d\boldsymbol{x} \\ &= \int \boldsymbol{x} \det U p_{\theta^*}(U^{-1}\boldsymbol{x}) d\boldsymbol{x} \\ &= \det U \int J_{U^{-1}}(\boldsymbol{x}) U \boldsymbol{x} p_{\theta^*}(\boldsymbol{x}) d\boldsymbol{x} \\ &= U \mathbb{E}_{p_{\theta^*}}[\boldsymbol{x}].\end{aligned}$$

The proof is complete. □

Theorem 3 guarantees that the estimated posterior distribution is equivalent to the true posterior up to a linear transformation. This result is formalized in the following corollary.

Corollary 1 (Latent Identifiability)**.** *Given that the assumptions of Theorem 7 are valid, the posterior distributions over latent variables $p_\theta(\boldsymbol{z}|\boldsymbol{x})$ are $\sim_P$-identifiable.*

Proof. Given that the ELBO criterion is optimized, the KL divergence term in (2.9) vanishes, and the variational posterior matches the model posterior which is consistent with the estimated likelihood and the prior, according to the Bayes rule.

Given that the Theorem 7 is valid, the combination of the true and inverse of the estimated nonlinearities $\boldsymbol{h} = \boldsymbol{f}^{-1} \circ f^*$, is a linear function, $\boldsymbol{h}(\boldsymbol{x}) = H\boldsymbol{x}$. Substituting $\boldsymbol{h}$ into (4.12), gives $A^{*+}HA\boldsymbol{z} \in \overline{\Delta}^M$ with $\boldsymbol{z} \in \overline{\Delta}^M$. Therefore, $A^{*+}HA = \Pi$ is a permutation matrix, and the true and estimated posterior distributions are $\sim_P$-equivalent.

□

In practice, the latent variables $\boldsymbol{z}$ can be estimated by sampling the posterior distribution. By Corollary 1 and Lemma 1, computing the mean of the posterior sample yields the expected value of the true latent variables up to a permutation.

4.3 Technical Lemmas

Here, we prove the technical lemmas that support the results discussed earlier.

Lemma 2. *Suppose $\boldsymbol{\theta} = (\boldsymbol{f}, \boldsymbol{A})$ and $\boldsymbol{\theta}^* = (\boldsymbol{f}^*, A^*)$ are the approximated and true parameters of the model* (4.6)*. If the marginal distributions are equal,*

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = p_{\boldsymbol{\theta}^*}(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in \mathcal{X}$$

then the noiseless distributions are equal almost everywhere:

$$\tilde{p}_{\boldsymbol{\theta}}(\boldsymbol{x}) = p(A^+\boldsymbol{f}^{-1}(\boldsymbol{x}))\,\mathrm{vol}(\boldsymbol{A})^{-1}\,\mathrm{vol}(J_{\boldsymbol{f}^{-1}}(\boldsymbol{x}))\mathbb{1}_{\mathcal{X}}(\boldsymbol{x})$$

Proof. By leveraging (B.2) and the change of variables $\bar{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{A}\boldsymbol{z})$, given Assumptions 7.1 and 7.2 are valid, we can rewrite the integral in the marginal probability density (4.6) as:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int\limits_{\mathcal{X}} \phi_\sigma(\boldsymbol{x} - \bar{\boldsymbol{x}}) p(A^+ \boldsymbol{f}^{-1}(\bar{\boldsymbol{x}})) \operatorname{vol}(\boldsymbol{A})^{-1} \operatorname{vol}(J_{\boldsymbol{f}^{-1}}(\bar{\boldsymbol{x}})) d\mu(\bar{x}),$$

where $\mathcal{X}$ is the image of Δ^M under $\boldsymbol{f} \circ \boldsymbol{A}$. Given that $\boldsymbol{A}$ is the full-rank matrix with independent columns, $A^+ = (A^\top \boldsymbol{A})^{-1} A^\top$. Introducing the shorthand notation for the transformed distribution,

$$\tilde{p}_{\boldsymbol{f},\boldsymbol{A}}(\boldsymbol{x}) = p(A^+ \boldsymbol{f}^{-1}(\boldsymbol{x})) \operatorname{vol}(\boldsymbol{A})^{-1} \operatorname{vol}(J_{\boldsymbol{f}^{-1}}(\boldsymbol{x})) \mathbb{1}_{\mathcal{X}}(\boldsymbol{x}),$$

we can write:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int\limits_{\mathbb{R}^N} \phi_\sigma(\boldsymbol{x} - \bar{\boldsymbol{x}}) \tilde{p}_{\boldsymbol{f},\boldsymbol{A}}(\bar{\boldsymbol{x}}) d\mu(\bar{\boldsymbol{x}}). \tag{4.15}$$

By applying the Fourier transform on both sides, and using the convolution property, we obtain:

$$\mathcal{F}[p_{\boldsymbol{\theta}}](\boldsymbol{\omega}) = \mathcal{F}[\tilde{p}_{\boldsymbol{f},\boldsymbol{A}}](\boldsymbol{\omega}) \mathcal{F}[\phi_\sigma](\boldsymbol{\omega}), \text{ for all } \boldsymbol{\omega}, \tag{4.16}$$

where $\mathcal{F}[\phi_\sigma](\boldsymbol{\omega}) = e^{-\frac{1}{2}\sigma^2 \|\boldsymbol{\omega}\|_2^2} \neq 0$ for all $\boldsymbol{\omega}$.

We have

$$p_\theta(\boldsymbol{x}) = p_{\theta^*}(\boldsymbol{x}). \tag{4.17}$$

By Assumption 7.4, $\mathcal{F}[\phi_\sigma](\omega)$ is non-zero almost everywhere, as a result equation (4.17) implies that

$$\mathcal{F}[\tilde{p}_{\boldsymbol{f},\boldsymbol{A}}](\boldsymbol{\omega}) = \mathcal{F}[\tilde{p}_{\boldsymbol{f}^*,A^*}](\boldsymbol{\omega}). \tag{4.18}$$

Given the uniqueness of the Fourier transform, according to Fact 2, it follows that:

$$\tilde{p}_{\boldsymbol{f},\boldsymbol{A}}(x) = \tilde{p}_{\boldsymbol{f}^*,A^*}(x) \tag{4.19}$$

This result indicates that the noise-free distributions must be identical if the noisy distributions are identical. □

Fact 2 (Proposition 3.8.6 [10])**.** *If two bounded Borel measures have equal Fourier transforms, then they coincide. In particular, two integrable functions with equal Fourier transforms are equal almost everywhere.*

Lemma 3. *If the following equation holds*

$$\mathbb{1}_{\overline{\operatorname{conv}}(\boldsymbol{A})}(\boldsymbol{y}) = \mathbb{1}_{\overline{\operatorname{conv}}(B)}(\boldsymbol{h}(\boldsymbol{y})), \tag{4.20}$$

then

$$\mathbf{1}^\top B^+ \boldsymbol{h}(A\boldsymbol{z}) = 1, \quad \forall \boldsymbol{z} \in \overline{\Delta}^M. \tag{4.21}$$

Proof. Equation

$$\mathbb{1}_{\overline{\text{conv}}(\boldsymbol{A})}(\boldsymbol{y}) = \mathbb{1}_{\overline{\text{conv}}(B)}(\boldsymbol{h}(\boldsymbol{y})), \quad \forall \boldsymbol{y} \in \mathbb{R}^N. \tag{4.22}$$

implies that $\boldsymbol{h}(\boldsymbol{y})$ is in the convex hull of A^*, if and only if $\boldsymbol{y}$ is in the convex hull of A^*. In other words,

$$\boldsymbol{y} \in \overline{\text{conv}}(\boldsymbol{A}) \leftrightarrows \boldsymbol{h}(\boldsymbol{y}) \in \overline{\text{conv}}(B), \quad \forall \boldsymbol{y} \in \mathbb{R}^N. \tag{4.23}$$

The two sides in (4.23) are equivalent to the following conditions:

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{z}, \quad \boldsymbol{z} \in \overline{\Delta}^M, \tag{4.24}$$

$$\boldsymbol{h}(\boldsymbol{y}) = A^*\boldsymbol{z}^*, \quad \boldsymbol{z}^* \in \overline{\Delta}^M, \tag{4.25}$$

and we can rewrite it as

$$\boldsymbol{z} \in \Delta^M \leftrightarrows B^+\boldsymbol{h}(A\boldsymbol{z}) \in \overline{\Delta}^M. \tag{4.26}$$

As a result, we obtain a functional equation

$$\mathbf{1}^\top B^+\boldsymbol{h}(A\boldsymbol{z}) = 1, \quad \forall \boldsymbol{z} \in \overline{\Delta}^M. \tag{4.27}$$

□

Fact 3 (Lemma 1 [50])**.** *Consider the Khatri–Rao (column-wise Kronecker) product defined as*

$$B \circledast A := [\boldsymbol{b}_1 \otimes \boldsymbol{a}_1, \cdots, \boldsymbol{b}_N \otimes \boldsymbol{a}_N],$$

where $A \in \mathbb{R}^{K\times N}$, $B \in \mathbb{R}^{L\times N}$, $\otimes$ stands for the Kronecker product, and $\boldsymbol{b}_n$, $\boldsymbol{a}_n$ are the columns of B and A. If the following condition holds:

$$\text{krank}(A) + \text{krank}(B) \geq N + 1,$$

then the matrix $B \circledast A$ has full column rank N.

Lemma 4. *Assume that Assumptions 7.1, 7.2, and 7.3 of the Theorem 7 are valid. If the following functional equation holds,*

$$\boldsymbol{b}\boldsymbol{h}(A\boldsymbol{z}) = 1, \quad \forall \boldsymbol{z} \in \overline{\Delta}^M, \tag{4.28}$$

$\boldsymbol{h}(\boldsymbol{z})$ is almost surely a linear transformation for $\boldsymbol{z} \in \overline{\Delta}^M$.

Proof. We first use $z_M = 1 - \sum_{i=1}^{M-1} z_i$ to rewrite the functional equation (4.28) as

$$\sum_{n=1}^{N} b_n\, h_n \left(\sum_{m=1}^{M-1} (a_{nm} - a_{nM}) z_m + a_{nM} \right) = 1.$$

By Fact 1, we know that for $\boldsymbol{z} \in \overline{\Delta}^M$, $\frac{\partial z_i}{\partial z_j} = 0$ for $i, j = 1, \ldots, M-1$ and $i \neq j$. Hence, differentiating with respect to z_l (where $l \in [M-1]$), and using the chain rule, this becomes:

$$\sum_{n=1}^{N} b_n(a_{nl} - a_{nM})h'_n \left(\sum_{m=1}^{M-1} (a_{nm} - a_{nM})z_m + a_{nM} \right) = 0, \tag{4.29}$$

Taking the second derivative with respect to z_k (for $k \in [M-1]$):

$$\sum_{n=1}^{N} b_n(a_{nl} - a_{nM})(a_{nk} - a_{nM})h''_n \left(\sum_{m=1}^{M-1} (a_{nm} - a_{nM})z_m + a_{nM} \right) = 0. \tag{4.30}$$

Now, we can express the system of equations in (4.30) in matrix form:

$$b \circledast G\boldsymbol{h}'' = \mathbf{0}, \tag{4.31}$$

where $\boldsymbol{h}'' = \begin{bmatrix} h''_1 & h''_2 & \cdots & h''_M \end{bmatrix}^\top$, and G is the matrix with dimensions $M(M-1)/2 \times N$, constructed as follows:

$$G = \begin{bmatrix} (\bar{\boldsymbol{a}}_1 \odot \bar{\boldsymbol{a}}_1)^\top \\ (\bar{\boldsymbol{a}}_2 \odot \bar{\boldsymbol{a}}_2)^\top \\ \vdots \\ (\bar{\boldsymbol{a}}_{M-1} \odot \bar{\boldsymbol{a}}_{M-1})^\top \\ (\bar{\boldsymbol{a}}_1 \odot \bar{\boldsymbol{a}}_2)^\top \\ \vdots \\ (\bar{\boldsymbol{a}}_{M-2} \odot \bar{\boldsymbol{a}}_{M-1})^\top \end{bmatrix}, \tag{4.32}$$

where each vector $\bar{\boldsymbol{a}}_i$ is defined as:

$$\bar{\boldsymbol{a}}_l = \begin{bmatrix} a_{1l} - a_{1N}, & a_{2l} - a_{2N}, & \ldots, & a_{Nl} - a_{NM} \end{bmatrix}^\top, \tag{4.33}$$

with $l = 1, \ldots, M-1$, and $\bar{A} = [\bar{\boldsymbol{a}}_1, \ldots, \bar{\boldsymbol{a}}_{M-1}]$.

If $b \circledast G$ has full column rank, then (4.31) yields $\boldsymbol{h}''(A\boldsymbol{z}) = 0$, which implies that $\boldsymbol{h}$ must be an affine function. To demonstrate it, we first establish that $\text{rank}(G) = N$ by identifying an $N \times N$ full-rank submatrix of G. As shown in the proof of Theorem 3 in Section 3.2.3, G almost surely has full Kruskal rank $\min(M(M-1), N)$. According to Fact 3, the Khatri-Rao product $b \circledast G$ achieves full column rank provided that $\text{krank}(b) + \text{krank}(G) \geq N + 1$. The rank of b is 1, as it is a non-zero vector, therefore $b \circledast G$ has full column rank, if $\text{krank}(G) \geq N$, i.e. $M(M-1)/2 \geq N$, which is satisfied by Assumption 7.2.

Finally, given that $\boldsymbol{h}$ is affine we can follow the arguments in the last part proof of Theorem 3 in Section 3.2.3, to conclude that $\boldsymbol{h}$ is a linear transformation.

□

4.4 Algorithm Design

In this section, we define the learning criterion and the optimization scheme to remove the nonlinear transformations and identify the latent components in model (4.1). We leverage a variational posterior from the logistic-normal family and optimize the ELBO loss using the VAE architecture, extending the optimization algorithm of VASCA [37]. This posterior belongs to the location-scale family, generated by the additive logistic transformation $g(\cdot) = \mathrm{softmax}([\cdot, 0])$, and is parameterized by the encoder neural networks $\boldsymbol{\sigma}(\boldsymbol{x}) : \mathbb{R}^N \to \mathbb{R}^{M-1}$ and $\boldsymbol{\mu}(\boldsymbol{x}) : \mathbb{R}^N \to \mathbb{R}^{M-1}$:

$$q_\phi(\boldsymbol{z}) = \frac{1}{\sqrt{2\pi}} \left(\prod_{m=1}^{M} z_m \right)^{-1} |\,\mathrm{diag}(\boldsymbol{\sigma}(\boldsymbol{x}))|^{-1\,{}^1\!/_2} \exp\left(-\frac{1}{2} \tilde{\boldsymbol{z}}^\top \mathrm{diag}(\boldsymbol{\sigma}(\boldsymbol{x}))^{-1} \tilde{\boldsymbol{z}} \right), \tag{4.34}$$

where $\tilde{z} = \log\left(\frac{\boldsymbol{z}_{-M}}{z_M}\right) - \mu(\boldsymbol{x})$ with $\boldsymbol{z}_{-M} = [z_1, \dots, z_{M-1}]$ and $z_M = 1 - \sum_{m=1}^{M-1} z_m$, the location parameter is given by the Gaussian mean and the diagonal scale matrix is the covariance matrix.

The difference with VASCA architecture lies in using a nonlinear decoder to model the post-nonlinear distortions. Specifically, our decoder consists of a trainable linear transformation followed by component-wise nonlinear distortions, parameterized by neural networks $f_n(x)$, $n = 1, \dots, N$, and is optimized jointly with the encoder using Adam optimizer [34].

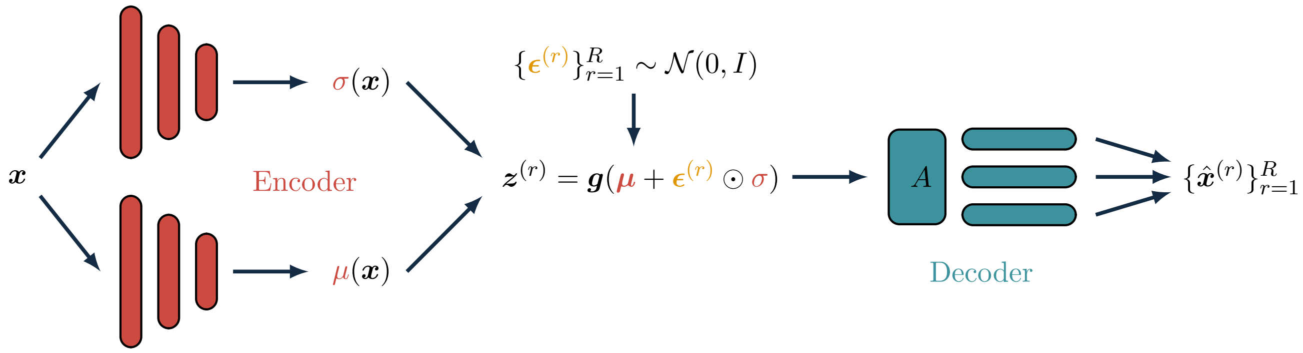
$\boldsymbol{x}$
$\sigma(\boldsymbol{x})$
Encoder
$\mu(\boldsymbol{x})$
$\{\boldsymbol{\epsilon}^{(r)}\}_{r=1}^R \sim \mathcal{N}(0, I)$
$\boldsymbol{z}^{(r)} = \boldsymbol{g}(\boldsymbol{\mu} + \boldsymbol{\epsilon}^{(r)} \odot \sigma)$
A
Decoder
$\{\hat{\boldsymbol{x}}^{(r)}\}_{r=1}^R$

Figure 4.1: This figure illustrates the architecture of the variational autoencoder for the Post-Nonlinear Simplex Component Analysis (NISCA) model. The notation follows Figure 2.3. In this configuration, the decoder includes a linear layer A, which represents the linear mixing, followed by neural networks that apply nonlinear distortions to each component. The transformation function $g(\cdot) = \mathrm{softmax}([\cdot, 0])$ is an additive logistic transformation, facilitating simplex-structured posterior and prior distributions.

We define the ELBO according to (2.12), where we drop the constant terms and apply normalization, so that the loss has a well-behaved noiseless limit. This yields the following minimization objective:

$$\ell(\theta, \phi; \boldsymbol{x}) = \sum_{r=1}^{R} \ell^{rec}(\theta; \boldsymbol{x}, \boldsymbol{z}^{(r)}) + \sigma_v^2 h_\phi(\boldsymbol{z}^{(r)}), \tag{4.35}$$

where $\boldsymbol{z}^{(r)}$ is sampled according to

$$\boldsymbol{z}^{(r)} = g(\boldsymbol{\mu}(\boldsymbol{x}) + \boldsymbol{\sigma}(\boldsymbol{x}) \odot \boldsymbol{\epsilon}^{(r)}), \quad \boldsymbol{\epsilon}^{(r)} \sim \mathcal{N}(0, \boldsymbol{I}).$$

The reconstruction loss and the pointwise entropy terms are defined as:

$$\ell^{rec}(\boldsymbol{f}, A; \boldsymbol{x}, \boldsymbol{z}) = \|\boldsymbol{x} - \boldsymbol{f}(A\boldsymbol{z})\|^2, \tag{4.36}$$

$$h_\phi(\boldsymbol{z}) = \tilde{z}^\top \operatorname{diag}(\boldsymbol{\sigma}(\boldsymbol{x}))^{-1}\tilde{z} + \sum_{i=1}^{M-1} \log \sigma_i(\boldsymbol{x}) + 2\sum_{i=1}^{M} \log z_i. \tag{4.37}$$

This loss function corresponds with the architecture as shown in Figure 4.1, and is optimized by Algorithm 1.

Algorithm 1 Probabilistic post-nonlinear simplex component analysis.

```
Input: Data tensor $X$, latent dimension $M$, noise variance $\sigma_v^2$
Initialize: Parameters $A$, $\boldsymbol{f}$, $\boldsymbol{\mu}$, and $\boldsymbol{\sigma}$
for epoch $= 1, \ldots,$ num_epochs do
    for $i = 1, \ldots,$ num_batches do
        $X_i \leftarrow$ i-th minibatch of $X$
        Draw $R$ samples $\{\boldsymbol{\epsilon}^{(r)}\}_{r=1}^{R}$ from $\mathcal{N}(0, \boldsymbol{I})$
        Compute $\boldsymbol{z}^{(r)} = \boldsymbol{g}(\boldsymbol{\mu}(X_i) + \boldsymbol{\epsilon}^{(r)} \odot \boldsymbol{\sigma}(X_i))$
        Compute reconstructed samples $\hat{X}_i^{(r)} = \boldsymbol{f}(A\boldsymbol{z}^{(r)})$
        Calculate loss $\frac{1}{R}\sum_{r=1}^{R} \ell^{(r)}(A, \boldsymbol{f}, \boldsymbol{\mu}, \boldsymbol{\sigma}; X_i, \hat{X}_i^{(r)})$
        Update $A$, $\boldsymbol{f}$, $\boldsymbol{\mu}$, and $\boldsymbol{\sigma}$ using Adam optimizer
    end for
end for
Output: Optimized parameters $A$, $\boldsymbol{f}$, $\boldsymbol{\mu}$, and $\boldsymbol{\sigma}$
```

In contrast to deterministic nonlinear approaches, our loss function incorporates an additional entropy term, which acts as a regularizer. This term encourages the approximate posterior distribution to remain close to the prior, thereby mitigating overfitting and preventing degenerate solutions, also known as the posterior collapse. By imposing this regularization, we ensure that the estimated posterior properly fills the latent space and captures an accurate approximation of the true posterior distribution.

Chapter 5: Experiments

In this chapter, we evaluate the performance of the proposed model through numerical experiments using synthetic data. Synthetic data allows for controlled generation of samples with specified nonlinear distortions and noise levels, enabling systematic validation of the model's theoretical guarantees. The availability of ground truth facilitates precise quantitative assessment of the estimated model parameters and latent components across various metrics. Additionally, access to the ground truth provides direct insight into essential model attributes, such as the dimensionality of the latent space.

The experiments are implemented in Python using the PyTorch Lightning framework on a MacBook Pro with an Apple M3 Pro chip, 12 cores at 4.05 GHz, and 18 GB of RAM. The source code is publicly available at `https://github.com/paukvlad/nisca`.

5.1 Experiment Design

5.1.1 Data Generation

The data is generated according to the model specified in (4.1). The mixing matrix A^* is sampled from a normal unit distribution, scaled by factor 10 to enhance the effect of the nonlinear distortions. Sampling from the normal distribution ensures linear independence of the columns. This is because the set of matrices with linearly dependent columns forms a lower-dimensional subspace within the space of all matrices, which has measure zero; thus, a randomly drawn matrix from a continuous distribution will lie outside this subspace with probability 1. The nonlinear functions f_n are chosen as variants of $\exp(\cdot)$, $\text{sigmoid}(\cdot)$, and $\tanh(\cdot)$ to ensure invertibility and reduce computational load. Latent components $\boldsymbol{z}^{(t)}$ are drawn from a uniform Dirichlet distribution $\mathcal{D}(\mathbf{1})$ and combined using the matrix A^*, followed by the nonlinear functions f_n, to produce noiseless observations $\boldsymbol{x}^{(t)} = \boldsymbol{f}(A\boldsymbol{z}^{(t)})$.

Gaussian noise is then added to the generated data to achieve the desired signal-to-noise ratio (SNR). The variance of the noise is set according to

$$\sigma_v^2 = \frac{1}{\text{SNR}} \sum_{t=1}^{T} \|\boldsymbol{x}^{(t)}\|_2^2,$$

where $\boldsymbol{v}^{(t)}$ is the noise vector for the t-th sample. This formulation maintains a noise level that scales with the total signal power, ensuring that the SNR remains consistent across all generated data samples. SNR is commonly expressed in decibels (dB), calculated as

$$\text{SNR}_{\text{dB}} = 10 \log_{10}(\text{SNR})\,\text{dB}.$$

For simplicity and ease of visualization, we restrict the model to two independent degrees of freedom, setting both the latent space dimension and the observed space dimension to $N = M = 3$. An example sample of the data vectors $\boldsymbol{x}$ is shown in Figure 5.1.

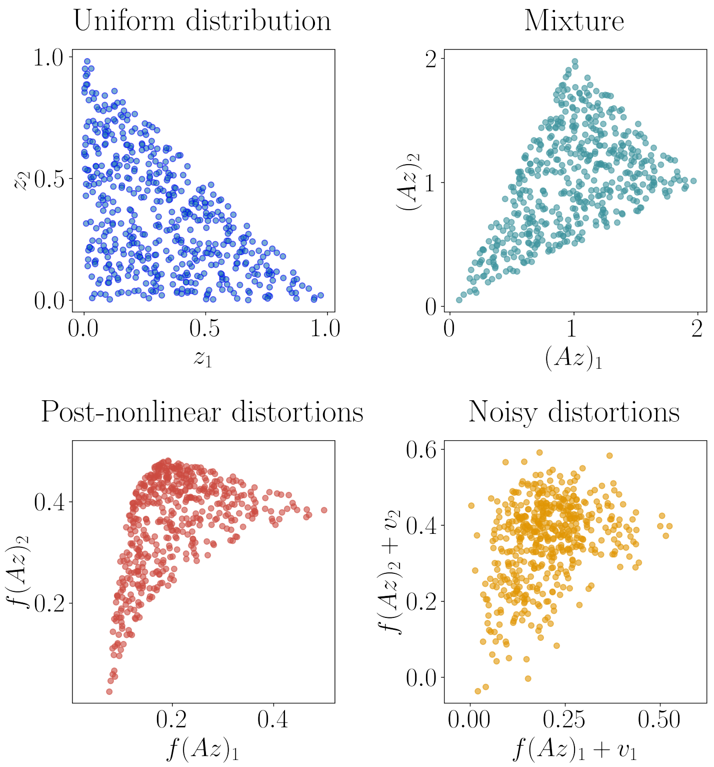

Figure 5.1: A sample of synthetic data generated by the model (4.1) with $N = M = 3$. The axes represent the two free degrees of freedom in the data.

5.1.2 Algorithm Settings

Unless otherwise specified the algorithm settings for our model remain consistent across all experiments. The decoder architecture utilizes an independent fully-connected neural network, or a multi-layer perceptron (MLP) for each observed component. In our simulation, a single-hidden-layer network with 128 neurons and ReLU activation achieves an effective balance between performance and computational efficiency. While increasing the network depth improves both performance and convergence speed, it also raises computational complexity. By contrast, increasing network width does not yield significant improvements. For more complex real-world experiments, optimizing the network architecture can be assisted by cross-validation and other deep learning techniques. Additionally, we include a single linear layer with $N = 3$ neurons, matching the target space dimension, and an input dimension of $M = 3$, without any activation function, to model the linear mixture of latent components that precedes the nonlinear transformations. The encoder is implemented as a fully-connected neural network with input dimension $N = 3$ matching the observed space dimension, and output dimension $M = 2$. Similarly to the decoder, a single-layer network with 128 neurons and ReLU activation is sufficient for our simulation settings.

To optimize parameters, we use the Adam optimizer [34] with a learning rate of 10^{-3} for the decoder and 10^{-2} for the encoder. The optimizer processes mini-batches of 100 data points, randomly sampled from the dataset, to estimate gradients for updating model parameters. Training runs for up to 5000 epochs, and stops if the latent MSE improvement between checks falls below 10^{-4} for 100 epochs.

5.1.3 Baselines and Benchmarks

VASCA serves as the baseline model, representing a linear version of our model. It employs the same encoder architecture, while the decoder is implemented as a single linear layer without activation to capture the linear mixture of latent components. The model is optimized using the Adam optimizer with learning rates identical to those of our model and a batch size of 1000 samples.

We use the constrained neural autoencoder (CNAE) [41] method as the primary benchmark for nonlinear experiments. The CNAE model only removes nonlinear distortions and requires subsequent application of the MVES algorithm to unmix latent components. The generated latent components, sufficiently scattered in Δ^{M-1} as defined in[17], are provably identifiable up to permutation ambiguities by MVES, provided the nonlinear distortions are removed. CNAE only considers the component-wise transformation with equal input and output dimensions. The decoder in this case is similar to ours, excepts for the linear layer, which is omitted. Both encoder and decoder have a single hidden layer with 128 neurons and ReLU activation functions. CNAE is trained using the augmented Lagrangian optimization, running up to 100 epochs of Adam with a constraint importance parameter $\rho = 10^2$, with the learning rate set to 10^{-3} for both the encoder and decoder, and the batch size set to 100 samples.

5.1.4 Metrics

The model is evaluated using three primary metrics. Nonlinearity removal is assessed via the *coefficient of determination*, denoted by R^2, commonly referred to as the R-square metric. It provides a measure of how well observed outcomes are replicated by the model, based on the proportion of total variation of outcomes explained by the model. Applied to $\boldsymbol{h} = \boldsymbol{f}^{-1} \circ \boldsymbol{f}^*$, it quantifies the alignment between the residual nonlinearity and its linear approximation, and provides a direct measure of the model's effectiveness in compensating for nonlinear distortions. Formally, the R^2-metric between $\boldsymbol{h}(\boldsymbol{y})$ and its linear fit $\tilde{\boldsymbol{h}} = C\boldsymbol{y}$, over the estimated linear mixtures $\boldsymbol{y}^{(t)} = A\boldsymbol{z}^{(t)}$ for $t = 1, \ldots, T$ is defined as

$$R^2 = 1 - \frac{\sum_{t=1}^{T} \|\tilde{h}^{(t)} - h^{(t)}\|_2^2}{\sum_{t=1}^{T} \|\tilde{h}^{(t)} - \bar{h}\|_2^2},$$

where $\bar{h} = \frac{1}{T}\sum_{i=1}^{T} \tilde{h}^{(t)}$. If the residual nonlinearity is perfectly straight, then $R^2 = 1$.

Another metric, that provides a measure of the model's ability to remove nonlinearity, is the *subspace distance* (SD) between the true and estimated latent spaces. By measuring nonlinear distortions in the latent space, SD provides a quantitative measure of the model's ability to remove nonlinearity under noisy conditions. Let $H = \boldsymbol{h}(AZ)$, where $Z = [\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(T)}]$. According to Theorem 7, $H = DAZ$, where D is a full rank diagonal matrix, thus H spans the same subspace as Z. The subspace distance is defined as

$$\operatorname{dist}(\mathcal{Z}, \hat{\mathcal{Z}}) = \|P_{\boldsymbol{z}}^{\perp} Q_{\boldsymbol{h}}\|_2,$$

with $\mathcal{Z} = \text{span}(Z^\top)$ and $\hat{\mathcal{Z}} = \text{span}(H^\top)$. Here, $Q_{\boldsymbol{h}}$ is the orthogonal basis of $\text{span}(H^\top)$ and $P_{\boldsymbol{z}}^\perp = I - Z(Z^\top Z)^{-1}Z^\top$ is the orthogonal projector onto the complement of $\text{span}(Z^\top)$. This performance metric, which is bounded within the interval $[0, 1]$, serves as an indicator of how well the subspace $\hat{\mathcal{Z}}$ approximates the subspace $\mathcal{Z}$. A value of 0 indicates perfect alignment between the subspaces, representing the best possible outcome, while larger values suggest greater deviations and, consequently, less effective nonlinearity removal.

The accuracy of latent space recovery is quantified by the mean square error (MSE) between the true and estimated latent variables.

$$\text{MSE} = \min_{\boldsymbol{\pi} \in \Pi_M} \frac{1}{M} \sum_{m=1}^{M} \left\| \boldsymbol{z}^*_{m,:} - \boldsymbol{z}_{\pi_m,:} \right\|_2^2, \tag{5.1}$$

where Π_M is the set of all permutations of $\{1, \ldots, M\}$, $\boldsymbol{z}^*_{m,:}$ and $\boldsymbol{z}_{m,:}$ are the ground truth and estimated m-th row of Z, respectively, and the m-th row in Z represents the m-th latent component across all samples. The permutation matrix reflects an intrinsic row permutation ambiguity in the estimated latent components that cannot be removed without additional prior knowledge.

5.2 Results

5.2.1 Linear Mixture

The experimental setup and algorithm details are as described in Section 5.1.2 and Section 5.1.1, except that the data generation does not include nonlinear distortions, and the batch size is 1000 for both VASCA and our model. First, we confirm that our model correctly identifies nonlinear distortions as linear functions, as shown in Figure 5.2. This result is expected, but nontrivial, as the decoder neural network is randomly initialized.

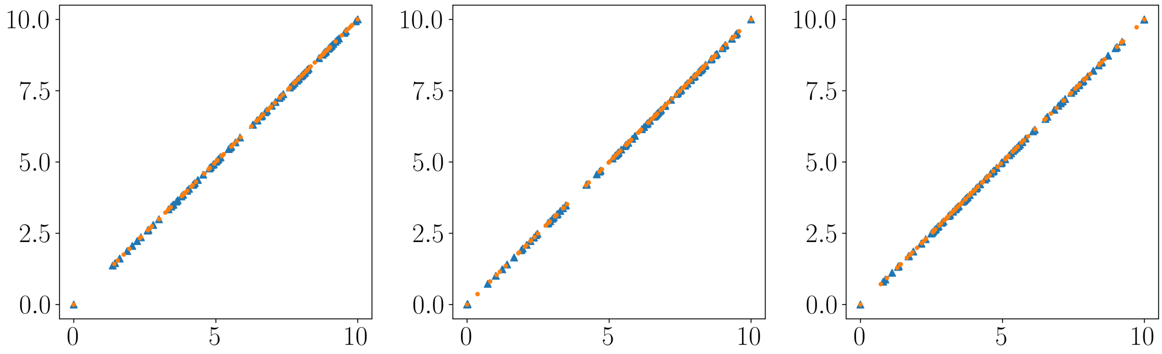

Figure 5.2: Nonlinearity removal in the linear case. Each plot shows the true (orange dots) and estimated (blue triangles) nonlinearities for each of the observed components (rescaled for better visualization).

Figure 5.3 demonstrates correlation between the true and the estimated latent components. The model correctly identifies the latent space and unmixes the latent components, providing a noisy approximation of the true latent variables, represented by the points scattered along the identity line.

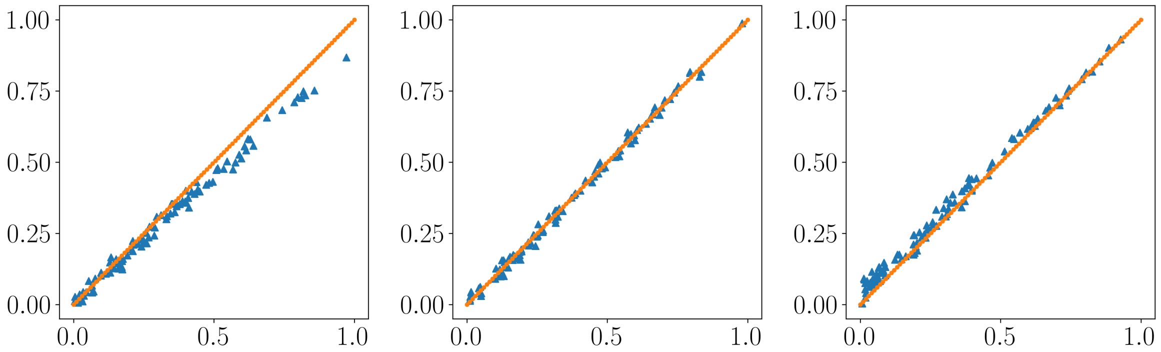

Figure 5.3: Latent component identification in the linear case with 30 dB SNR. Each plot shows correlation between the true and estimated latent component in a random sample of training data, with the true components residing on the x-axis and the estimated components on the y-axis. The orange line represents identity.

Table 5.1 and Table 5.2 show the MSE and SD for different noise levels, averaged over 5 random trials. The proposed method shows comparable performance to VASCA benchmark in terms of both metrics, and is correctly recovering the latent components. In contrast to VASCA, our model does not identify the mixture matrix. This is because of the ambiguity introduced by the learned linear post-mixture transformation in the decoder, which is identity in VASCA.

Table 5.1: MSE of the estimated latent components under various SNR in linear mixture.

Model	**10dB**	**20dB**	**30dB**
Proposed	$2.82 \cdot 10^{-2}$	$5.43 \cdot 10^{-3}$	$5.63 \cdot 10^{-3}$
VASCA	$2.93 \cdot 10^{-2}$	$5.47 \cdot 10^{-3}$	$5.03 \cdot 10^{-3}$

Table 5.2: SD between the estimated latent components under various SNR in linear mixture.

Model	**10dB**	**20dB**	**30dB**
Proposed	0.331	0.130	0.132
VASCA	0.357	0.131	0.128

5.2.2 Nonlinearity Removal

Next, we assess the model's ability to identify nonlinear distortions. The noise level is set to 30dB as before. The nonlinear distortions applied to each dimension are:

$$f_1(x_1) = 5\,\mathrm{sigmoid}(x_1) + 0.3x_1,$$
$$f_2(x_2) = 3\tanh(x_2)0.2x_2,$$
$$f_3(x_3) = 0.4\exp(x_3).$$

Note, that these functions are not disclosed to the learning algorithm and are used for testing and evaluation only. We run trainer for up to 5000 epochs, and use the same stopping criterion as in the linear case.

Figure 5.4 shows the true and estimated nonlinearities, scaled to fill the same range for better visualization. As we can see, nonlinearities learned by the proposed method visually align with the true distortions. The R^2 values are 0.9931, 0.9898, and 0.9912 for the first, second, and third components, respectively. The discrepancy between the true and estimated nonlinearities is attributed to the finite-sample effects.

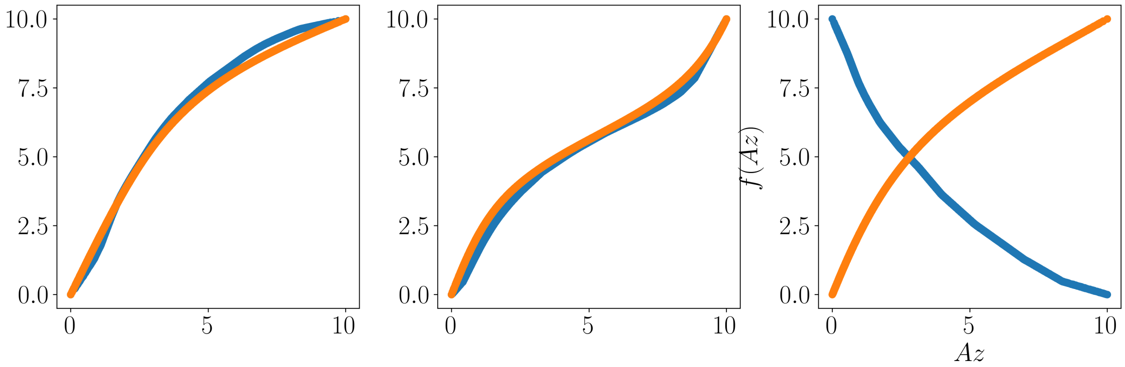

Figure 5.4: Nonlinearity removal in noisy post-nonlinear mixture data. Each plot shows the true (orange dots) and estimated (blue traingles) nonlinearities for each of the observed components (rescaled for better visualization).

5.2.3 Latent Space Identification

Figure 5.5 shows correlation between the true and estimated latent components, corresponding with the estimated nonlinearities in Figure 5.4. Each scatter plot shows the true latent component on the x-axis and the estimated latent component on the y-axis. The closer the points are to the identity line, the better is the unmixing. Misalignment means that the component

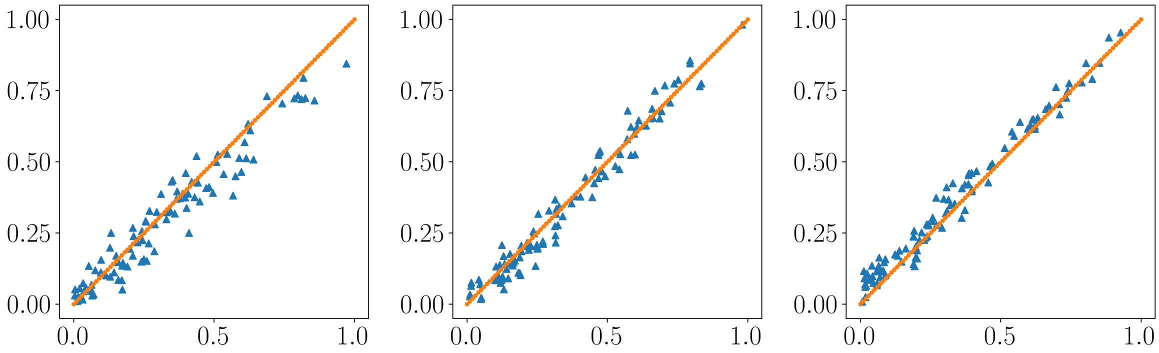

Figure 5.5: Latent component identification in the post-nonlinear mixture model with 30 dB SNR. Each plot shows correlation between the true and estimated latent component in a random sample of training data, with the true components residing on the x-axis and the estimated components on the y-axis. The orange line represents identity.

is scaled or shifted, while the spread of the points indicates the components were not entirely separated. The estimates are visually well aligned with the ground truth, demonstrating that our model correctly recovers the latent space and achieves unmixing in the presence of noise and nonlinear distortions. For the given experimental settings, our model yields the averaged latent MSE of $5.23 \cdot 10^{-3}$, with SD 0.131. Compared to the linear case, the values are higher, which is expected due to the presence of nonlinear distortions. Nevertheless, the model is able to unmix the latent components to a degree comparable to CNAE/MVES pipeline, which yields the SD of 0.123 for similar experimental settings. This is analogous to around only 2 degrees of misalignment between two vectors.

5.2.4 Impact of Noise

Finally, we systematically evaluate the impact of noise and compare the performance of our model with the benchmarks, under different SNR values. Settings are the same as before. The SNR is set to 10, 20, and 30 dB, and the results are averaged over 5 random trials. Table 5.3 shows that the nonlinear models by far outperform the baseline, especially at higher SNR levels. At lower SNR levels, the performance gap is smaller, as the noise dominates the signal, especially for the latent space metrics MSE and SD in Tables 5.4, and 5.5. At the same time, our model shows superior performance in removing nonlinear distortions in highly noisy conditions, as evidenced by higher R^2 values compared to the benchmark CNAE/MVES pipeline. The benchmark method exhibits a decline in nonlinearity removal effectiveness as noise levels increase, whereas our model

Table 5.3: R^2 of the estimated latent components under various SNR in nonlinear mixture.

Model	**10dB**	**20dB**	**30dB**
VASCA	0.671	0.561	0.715
CNAE/MVES	0.772	0.945	0.991
Proposed	0.897	0.971	0.994

Table 5.4: MSE of the estimated latent components under various SNR in nonlinear mixture.

Model	**10dB**	**20dB**	**30dB**
VASCA	$7.34 \cdot 10^{-2}$	$6.81 \cdot 10^{-2}$	$4.87 \cdot 10^{-2}$
CNAE/MVES	$6.85 \cdot 10^{-2}$	$2.07 \cdot 10^{-2}$	$5.01 \cdot 10^{-3}$
Proposed	$6.71 \cdot 10^{-2}$	$2.21 \cdot 10^{-2}$	$5.23 \cdot 10^{-3}$

Table 5.5: SD of the estimated latent components under various SNR in nonlinear mixture.

Model	**10dB**	**20dB**	**30dB**
VASCA	0.562	0.611	0.607
CNAE/MVES	0.518	0.226	0.123
Proposed	0.493	0.272	0.131

maintains robust performance across all SNR levels. We can also see that the efficiency of the latent space identification is similar to the linear case. This is expected when the nonlinear distortions are correctly identified, as the latent space is then estimated from the linear mixtures.

Chapter 6: Conclusion

6.1 Implications of Results

In this thesis, we revisited post-nonlinear simplex component analysis through the lens of model identifiability, and proposed an identifiable probabilistic model, termed the post-nonlinear simplex component analysis (NISCA). By leveraging the geometric properties of the probability simplex, we demonstrated that latent component identifiability is achievable even in the presence of post-nonlinear distortions and noise, thus enhancing the interpretability and practical utility of unsupervised simplex component analysis in real-world applications. Our approach has direct implications for a wide range of practical tasks, including audio and speech processing, hyperspectral imaging, topic modeling, medical imaging, and image classification. The main theoretical contributions of this work are Theorem 7 and Corollary 1, that guarantee recovery of the true generative process up to nonessential scale-permutation ambiguities under mild regularity conditions.

Existing approaches to the post-nonlinear simplex component analysis, such as the constrained nonlinear autoencoder (CNAE) [41] do not support out-of-the-box recovery of the latent space. The proposed NISCA model, on the other hand, provides a principled approach to latent space estimation, by incorporating the prior knowledge about the geometric structure of the latent space. Furthermore, NISCA handles noise in a probabilistic manner, and thus is more robust to noisy observations compared to deterministic methods. As a generative model, NISCA also enables data generation, by sampling from the learned latent space. This generative capability is particularly valuable in applications requiring controlled data augmentation or simulation, including engineering, financial modeling, scientific simulations, data visualization, and anomaly detection.

We evaluated the proposed model against VASCA [37], a probabilistic linear baseline, and CNAE/MVES pipeline [41], a deterministic nonlinear benchmark. Performance metrics included R^2 (R-squared) and subspace distance (SD) for assessing nonlinearity removal, as well as mean squared error (MSE) for latent space recovery. Our model demonstrates greater robustness to noise and provides a faster algorithm for nonlinearity removal. Notably, at low SNR values, NISCA outperforms the constrained nonlinear autoencoder (CNAE), the primary benchmark, in terms of the R^2 metric, as detailed in Table 5.3. The most significant improvement is observed at an SNR of 10 dB, where NISCA achieves an R^2 score of 0.90 compared to 0.77 for CNAE. In terms of latent component recovery, NISCA exhibits performance comparable to the benchmark pipeline, as evidenced by the MSE and SD metrics presented in Tables 5.4 and 5.5.

6.2 Limitations and Challenges

While our findings demonstrate the efficiency of NISCA in latent space estimation in simplex component analysis, this approach has notable limitations. First, our model incurs significant computational cost when the data feature dimension N is large, as each dimension requires training an individual neural network. Scaling the model to high-dimensional datasets is thus challenging without further optimization. Another computational bottleneck is due to latent space sampling. In our simulations, we bypassed this step, and instead relied on the mean of the posterior distribution for latent space estimation. Second, our approach is specifically designed for the data generation model (4.1). When nonlinear distortions extend beyond component-wise nonlinearities, this framework does not guarantee effective nonlinearity removal. Furthermore, the identifiability results rely on the assumptions of Gaussian noise and a Dirichlet-distributed prior, which may not always align with real-world data characteristics.

6.3 Future Directions

Based on the findings of this study, promising avenues for future research include the following subjects. To address the limitations of NISCA and extend the applicability of simplex-structured probabilistic modeling, more general geometric constraints on the prior, such as null space constraints, can be incorporated into the model, drawing ideas from deterministic post-nonlinear SCA model [42]. Another direction is to investigate the impact of latent space dimensionality on model performance. Since the dimension of the latent space is often unknown in real-world applications, it is essential to evaluate the model behaviour under varying dimensions and explore techniques like Bayesian model selection or cross-validation to infer dimensionality from data. To enhance the model's ability to handle more complex nonlinearities encountered in real-world datasets, it is essential to systematically analyze performance of network architectures and optimization strategies, using hyperparameter search and model selection techniques. Preliminary experiments suggest that the proposed model may be particularly effective with limited or corrupted data, pointing to another potential advantage of density estimation over point-wise estimation for latent components. To support this, finite-sample analysis, particularly in low SNR regimes, could provide a structured evaluation of the model's performance in practical scenarios. Finally, due to computational resource limitations, assessing the model on real hyperspectral data was beyond the scope of this thesis. Applications in areas such as hyperspectral unmixing, medical imaging, and financial forecasting are the key objectives for the forthcoming publications building on the foundation established in this thesis.

APPENDICES

Appendix A: Dirichlet Distribution

Definition 5. *Let $\boldsymbol{z} \in \Delta^{N-1}$ denote the $\boldsymbol{\alpha}$-Dirichlet distributed random variable. The density of $\overline{\boldsymbol{z}} = (z_1, \ldots, z_{N-1})$ is:*

$$\mathcal{D}(\overline{\boldsymbol{z}}; \boldsymbol{\alpha}) = \frac{1}{B(\boldsymbol{\alpha})} \left(\prod_{i=1}^{N-1} z_i^{\alpha_i - 1} \right) \left(1 - \sum_{i=1}^{N-1} z_i \right)^{\alpha_N - 1} \mathbb{1}_{\tilde{\Delta}^{N-1}}(\overline{\boldsymbol{z}}), \tag{A.1}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^N_{++}$ is the concentration parameter, $\tilde{\Delta}$ is the distribution support,

$$\tilde{\Delta}^{N-1} = \{\overline{\boldsymbol{z}} \in \mathbb{R}^{N-1}_{++} | 1 - \mathbf{1}^\top \overline{\boldsymbol{z}} > 0\}, \tag{A.2}$$

the multivariate beta function $B(\boldsymbol{\alpha})$ is given by

$$B(\boldsymbol{\alpha}) = \frac{\prod_{i=1}^N \Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^N \alpha_i)}$$

and $\Gamma(x) = \int_0^\infty t^{x-1} e^{-t}\, dt$ is the Gamma function.

It is commonly defined with respect to the Lebesgue measure on the Euclidean space $\mathbb{R}^{N-1}$,

$$\mathcal{D}(\boldsymbol{z}; \boldsymbol{\alpha}) = \frac{1}{B(\boldsymbol{\alpha})} \left(\prod_{i=1}^{N} z_i^{\alpha_i - 1} \right) \mathbb{1}_{\overline{\Delta}^{N-1}}(\boldsymbol{z})$$

where $\boldsymbol{z}$ belongs to the standard $(N-1)$-simplex, and write $\boldsymbol{z} \sim \mathcal{D}(\cdot; \boldsymbol{\alpha})$ to specify a Dirichlet random variable. When $\boldsymbol{\alpha} = \mathbf{1}$, we obtain the uniform unit-simplex distribution:

$$\mathcal{D}(\boldsymbol{z}; \mathbf{1}) = (N-1)! \cdot \mathbb{1}_{\overline{\Delta}^{N-1}}(\boldsymbol{z}). \tag{A.3}$$

Fact 4 (Dirichlet moments, see [44])**.** *Let $\boldsymbol{z} \sim \mathcal{D}(\cdot; \boldsymbol{\alpha})$.*

(a) The expectation of $\boldsymbol{z}$ is given by

$$\mathbb{E}[\boldsymbol{z}] = \tilde{\boldsymbol{\alpha}}, \tag{A.4}$$

where $\tilde{\boldsymbol{\alpha}} = \boldsymbol{\alpha}/\alpha_0$ and $\alpha_0 = \sum_{i=1}^N \alpha_i$.

(b) The covariance of $\boldsymbol{z}$ is

$$cov(\boldsymbol{z}) = \frac{1}{1 + \alpha_0} \left(diag(\tilde{\boldsymbol{\alpha}}) - \tilde{\boldsymbol{\alpha}} \tilde{\boldsymbol{\alpha}}^\top \right) \tag{A.5}$$

(c) The entropy of $\boldsymbol{z}$ is

$$H(\boldsymbol{z}) := \mathbb{E}[-\log \mathcal{D}(\boldsymbol{z}; \boldsymbol{\alpha})] = \log B(\boldsymbol{\alpha}) - \sum_{i=1}^N (\alpha_i - 1)(\psi(\alpha_i) - \psi(\alpha_0)) \tag{A.6}$$

where $\psi(x) = \log\Gamma(x)'$ *is the digamma function.*

Fact 5 (uniform distribution on a full-dimensional simplex)**.** *Let* $\boldsymbol{x} = \boldsymbol{B}\boldsymbol{z}$*, where* $\boldsymbol{B} \in \mathbb{R}^{(N-1)\times N}$ *is affinely independent and* $\boldsymbol{z} \sim \mathcal{D}(\cdot, \mathbf{1})$*. The PDF of* $\boldsymbol{x}$ *is given by:*

$$p(\boldsymbol{x}) = \frac{1}{\text{vol } \boldsymbol{B}} \mathbb{1}_{\overline{\text{conv}}(\boldsymbol{B})}(\boldsymbol{x}). \tag{A.7}$$

Proof. By $z_N = 1 - \mathbf{1}^\top \bar{\boldsymbol{z}}$, we can write,

$$\boldsymbol{x} = \sum_{i=1}^{N-1} \boldsymbol{b}_i z_i + \boldsymbol{b}_N \left(1 - \sum_{i=1}^{N-1} z_i\right) = \bar{\boldsymbol{B}}\bar{\boldsymbol{z}} + \boldsymbol{b}_N$$

where $\overline{\boldsymbol{B}}$ is invertible due to the affine independence of $\boldsymbol{B}$. Since the mapping from $\bar{\boldsymbol{z}}$ to $\boldsymbol{x}$ is bijective, we can apply transformation of random variables to obtain

$$p(\boldsymbol{x}) = \frac{1}{|\det \bar{\boldsymbol{B}}|} \mathcal{D}(\bar{\boldsymbol{B}}^{-1}(\boldsymbol{x} - \boldsymbol{b}_N); \mathbf{1}),$$

where D is defined in the Dirichlet distribution in (A.1). It can be verified that

$$\bar{\boldsymbol{B}}^{-1}(\boldsymbol{x} - \boldsymbol{b}_N) \in \tilde{\Delta} \Leftrightarrow \boldsymbol{x} \in \overline{\text{conv}}(\boldsymbol{B}),$$

and hence

$$\mathcal{D}(\bar{\boldsymbol{B}}^{-1}(\boldsymbol{x} - \boldsymbol{b}_N); \mathbf{1}) = (N-1)!\, \mathbb{1}_{\overline{\text{conv}}(\boldsymbol{B})}(\boldsymbol{x}).$$

Finally, from (2.1), we have vol $\boldsymbol{B} = |\det \bar{\boldsymbol{B}}|/(N-1)!$ when $\bar{\boldsymbol{B}}$ is square.

□

Appendix B: Change of Variables

Definition 6 (The Lebesgue integral)**.** *Let $f : \mathbb{R}^N \to \mathbb{R}$ be a measurable function. The Lebesgue integral of f over a set $\mathcal{X} \subseteq \mathbb{R}^N$ is defined as*

$$\int_{\mathcal{X}} f(\boldsymbol{x})\, d\mu(\boldsymbol{x}) = \int_{\mathbb{R}^N} \mathbb{1}_{\mathcal{X}}(\boldsymbol{x}) f(\boldsymbol{x})\, d\mu(\boldsymbol{x}), \tag{B.1}$$

where $\mathbb{1}_{\mathcal{X}}(\boldsymbol{x})$ is the indicator function of $\mathcal{X}$, and μ is the Lebesgue measure on $\mathbb{R}^N$.

The integral over the simplex Δ^{N-1} is given by

$$\int_{\Delta^{N-1}} f(\boldsymbol{z})\, d\mu(\boldsymbol{z}) = \int_{\mathbb{R}_+^{N-1}} f(\bar{\boldsymbol{z}}, 1 - \mathbf{1}^\top \bar{\boldsymbol{z}})\, d\bar{\boldsymbol{z}},$$

where we use the Lebesgue integral for a compact and symmetric representation.

The change of variables formula allows to transforming an integral over one set $\mathcal{Z} \subset \mathbb{R}^M$ into an integral over another set $\mathcal{X} \subset \mathbb{R}^N$ $(N > M)$ via a sufficiently well-behaved function $\phi : \mathcal{X} \to \mathcal{Z}$:

$$\int_{\mathcal{Z}} f(\boldsymbol{z})\, d\boldsymbol{z} = \int_{\mathcal{X}} (f \circ \phi)(\boldsymbol{x})\, \mathrm{vol}(J_\phi(\boldsymbol{x}))\, d\boldsymbol{x}, \tag{B.2}$$

where vol$(\cdot)$ denotes the matrix volume, J_ϕ is the full column rank Jacobian matrix of ϕ over $\mathcal{X}$,

$$J_\phi = \left(\frac{\partial \phi_i}{\partial x_j}\right),$$

and the integration measures $d\boldsymbol{z}$ and $d\boldsymbol{x}$ are defined with respect to the Lebesgue measure on $\mathcal{Z}$ and $\mathcal{X}$, respectively. The volume of a matrix is defined as the square root of the sum of the squares of the determinants of all possible maximal square submatrices (submatrices of full rank)

$$\mathrm{vol}(A) = \sqrt{\sum_{(I,J) \in \mathcal{N}(A)} (\det A_{IJ})^2}, \tag{B.3}$$

where the sum is taken over the index set $\mathcal{N}(A)$ of all $r \times r$ nonsingular submatrices A_{IJ}. The matrix volume is the generalization of the absolute value of determinant from nonsingular to arbitrary matrices. Alternatively, it can be viewed as the product of the singular values of the matrix. For a full column rank matrix, this volume simplifies to:

$$\mathrm{vol}(A) = \sqrt{\det(A^T A)}, \tag{B.4}$$

and for a square matrix, $\mathrm{vol}(A) = |\det(A)|$.

According to the inverse function theorem, the inverse of the Jacobian of a function is the

Jacobian of the inverse function,

$$\boldsymbol{J}_{f^{-1}}(\boldsymbol{x}) = (\boldsymbol{J}_f(f^{-1}(\boldsymbol{x})))^{-1}. \tag{B.5}$$

Appendix C: Fourier Transformation

For $f \in L^1(E^n)$, the Fourier transform (FT) $\mathcal{F}[f]$: $E^n \rightarrow E$ is defined as:

$$\mathcal{F}[f](\boldsymbol{\xi}) = \int\limits_{E^n} f(\boldsymbol{x}) e^{-2\pi i \boldsymbol{\xi}^\top \boldsymbol{x}} \, d\boldsymbol{x}, \tag{C.1}$$

where $\xi \in E^n$. The inverse FT is given by:

$$f(\boldsymbol{x}) = \frac{1}{2\pi} \int\limits_{E^n} \mathcal{F}[f](\boldsymbol{\xi}) e^{i2\pi \boldsymbol{\xi}^\top \boldsymbol{x}} \, d\boldsymbol{\xi}. \tag{C.2}$$

A convolution of two functions $f, g \in L^1(E^n) : E^n \rightarrow E$ is defined as:

$$(f * g)(\boldsymbol{x}) = \int\limits_{E^n} f(\boldsymbol{x} - \boldsymbol{y}) g(\boldsymbol{y}) \, d\boldsymbol{y}. \tag{C.3}$$

The FT of a convolution is the product of the FTs of the functions (Theorem 1.4 in [51]):

$$\mathcal{F}[f * g](\boldsymbol{\xi}) = \mathcal{F}[f](\boldsymbol{\xi}) \mathcal{F}[g](\boldsymbol{\xi}). \tag{C.4}$$

We say that $f \in L^1(E^n)$ is Gauss summable to l if

$$\lim_{\varepsilon \to 0} G_\varepsilon(f) = \lim_{\varepsilon \to 0} \int\limits_{E^n} f(\boldsymbol{x}) e^{-\varepsilon \|\boldsymbol{x}\|^2} \, d\boldsymbol{x} = l. \tag{C.5}$$

exists and equals l.

It can be put the in the form

$$M_{\varepsilon,\Phi}(f) = M_\varepsilon(f) = \int\limits_{E^n} \Phi(\varepsilon \boldsymbol{x}) f(\boldsymbol{x}) \, d\boldsymbol{x} \tag{C.6}$$

where $\Phi \in C_0$ and $\Phi(0) = 1$. Then $\int_{E_n} f$ is summable to l if

$$\lim_{\varepsilon \to 0} M_\varepsilon(f) = l. \tag{C.7}$$

We shall call $M_\varepsilon(f)$ the Φ means of this integral.

The Gauss-Weierstrass (GW) kernel $\phi_\sigma(\boldsymbol{z})$ is defined as:

$$\phi_\sigma(\boldsymbol{z}) = (2\pi\sigma^2)^{-n/2} e^{-\frac{\|\boldsymbol{z}\|^2}{2\sigma^2}}, \tag{C.8}$$

$$\phi_\sigma(\alpha) = (4\pi\alpha)^{-n/2} e^{-\frac{\|\boldsymbol{z}\|^2}{4\alpha}}, \tag{C.9}$$

where $\sigma > 0$ is the standard deviation. The FT of the GW kernel is given by (Theorem 1.13 in [51]):

$$\mathcal{F}[\phi_\sigma](\boldsymbol{\xi}) = \int e^{-i2\pi\boldsymbol{\xi}^\top \boldsymbol{n}}\phi_\sigma(\boldsymbol{n})d\mu(\boldsymbol{n}) = e^{-\frac{1}{2}\sigma^2\|\boldsymbol{\xi}\|^2} \tag{C.10}$$

Fact 6 ([51], Theorem 1.16)**.** *If $f : \mathbb{R}^n \to \mathbb{R}$ belongs to $L^1(\mathbb{R}^n)$, the space of all measurable functions defined on $\mathbb{R}^n$ and with $\int_{\mathbb{R}^n} \|f(\boldsymbol{x})\|_1 \, d\boldsymbol{x} < \infty$ ($\|\cdot\|_1$ denotes the 1-norm), then*

$$\int\limits_{\mathbb{R}^n} \mathcal{F}[\phi_\varepsilon](\boldsymbol{\xi})f(\boldsymbol{\xi})e^{i2\pi\boldsymbol{\xi}^\top \boldsymbol{z}}\, d\boldsymbol{\xi} = \int\limits_{\mathbb{R}^n} \phi_\varepsilon(\boldsymbol{z}-\boldsymbol{x})f(\boldsymbol{x})\, d\boldsymbol{x} \tag{C.11}$$

for all $\varepsilon > 0$.

Bibliography

[1] Gaussian Mixture Model - an overview | ScienceDirect Topics. https://www.sciencedirect.com/topics/mathematics/gaussian-mixture-model.

[2] Learning the parts of objects by non-negative matrix factorization | Nature. https://www.nature.com/articles/44565.

[3] Nonlinear Unmixing of Hyperspectral Images: Models and Algorithms | IEEE Journals & Magazine | IEEE Xplore. https://ieeexplore.ieee.org/document/6678284.

[4] Numerical Optimization | SpringerLink. https://link.springer.com/book/10.1007/978-0-387-40065-5.

[5] A Signal Processing Perspective on Hyperspectral Unmixing: Insights from Remote Sensing | IEEE Journals & Magazine | IEEE Xplore. https://ieeexplore.ieee.org/document/6678258.

[6] S. Achard and C. Jutten. Identifiability of post-nonlinear mixtures. *IEEE Signal Processing Letters*, 12(5):423–426, May 2005.

[7] Yoann Altmann, Nicolas Dobigeon, and Jean-Yves Tourneret. Unsupervised Post-Nonlinear Unmixing of Hyperspectral Images Using a Hamiltonian Monte Carlo Algorithm. *IEEE Transactions on Image Processing*, 23(6):2663–2675, June 2014.

[8] Shunichi Amari. A Theory of Adaptive Pattern Classifiers. *IEEE Transactions on Electronic Computers*, EC-16(3):299–307, June 1967.

[9] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, New York, 2006.

[10] V. I. Bogachev. *Measure Theory*. Springer, Berlin ; New York, 2007.

[11] Richard Caron. The Zero Set of a Polynomial, 2005.

[12] Pierre Comon. Independent component analysis, A new concept? *Signal Processing*, 36(3):287–314, April 1994.

[13] Sergio Cruces. Bounded Component Analysis of Linear Mixtures: A Criterion of Minimum Convex Perimeter. *IEEE Transactions on Signal Processing*, 58(4):2141–2154, April 2010.

[14] Yannick Deville. From separability/identifiability properties of bilinear and linear-quadratic mixture matrix factorization to factorization algorithms. *Digital Signal Processing*, 87:21–33, April 2019.

[15] Nicolas Dobigeon, SaÏd Moussaoui, Martial Coulon, Jean-Yves Tourneret, and Alfred O. Hero. Joint Bayesian Endmember Extraction and Linear Unmixing for Hyperspectral Imagery. *IEEE Transactions on Signal Processing*, 57(11):4355–4368, November 2009.

[16] Denis G. Fantinato, Leonardo T. Duarte, Yannick Deville, Romis Attux, Christian Jutten, and Aline Neves. A second-order statistics method for blind source separation in post-nonlinear mixtures. *Signal Processing*, 155:63–72, February 2019.

[17] Xiao Fu, Kejun Huang, Nicholas D. Sidiropoulos, and Wing-Kin Ma. Nonnegative Matrix Factorization for Signal and Data Analytics: Identifiability, Algorithms, and Applications. *IEEE Signal Processing Magazine*, 36(2):59–80, March 2019.

[18] Xiao Fu, Kejun Huang, Bo Yang, Wing-Kin Ma, and Nicholas D. Sidiropoulos. Robust Volume Minimization-Based Matrix Factorization for Remote Sensing and Document Clustering, August 2016.

[19] Xiao Fu, Kejun Huang, Bo Yang, Wing-Kin Ma, and Nicholas D. Sidiropoulos. Robust Volume Minimization-Based Matrix Factorization for Remote Sensing and Document Clustering. *IEEE Transactions on Signal Processing*, 64(23):6254–6268, December 2016.

[20] Xiao Fu, Wing-Kin Ma, Kejun Huang, and Nicholas D. Sidiropoulos. Blind Separation of Quasi-Stationary Sources: Exploiting Convex Geometry in Covariance Domain. *IEEE Transactions on Signal Processing*, 63(9):2306–2320, May 2015.

[21] Nicolas Gillis and Stephen A. Vavasis. Fast and Robust Recursive Algorithms for Separable Nonnegative Matrix Factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(4):698–714, April 2014.

[22] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks, June 2014.

[23] Michael U Gutmann, Michael Gutmann, Aapo Hyvarinen, and Aapo Hyvarinen. Noise-Contrastive Estimation of Unnormalized Statistical Models, with Applications to Natural Image Statistics.

[24] Kejun Huang and Xiao Fu. Detecting Overlapping and Correlated Communities without Pure Nodes: Identifiability and Algorithm. In *Proceedings of the 36th International Conference on Machine Learning*, pages 2859–2868. PMLR, May 2019.

[25] Aapo Hyvärinen, Juha Karhunen, and Erkki Oja. *Independent Component Analysis*. J. Wiley, New York, 2001.

[26] Aapo Hyvärinen, Ilyes Khemakhem, and Ricardo Monti. Identifiability of latent-variable and structural-equation models: From linear to nonlinear. *Annals of the Institute of Statistical Mathematics*, 76(1):1–33, February 2024.

[27] Aapo Hyvärinen and Hiroshi Morioka. Unsupervised Feature Extraction by Time-Contrastive Learning and Nonlinear ICA. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

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

[29] Aapo Hyvärinen, Hiroaki Sasaki, and Richard Turner. Nonlinear ICA Using Auxiliary Variables and Generalized Contrastive Learning. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, pages 859–868. PMLR, April 2019.

[30] Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An Introduction to Variational Methods for Graphical Models. In Michael I. Jordan, editor, *Learning in Graphical Models*, pages 105–161. Springer Netherlands, Dordrecht, 1998.

[31] Christian Jutten and Jeanny Herault. Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture. *Signal Processing*, 24(1):1–10, July 1991.

[32] Ilyes Khemakhem, Diederik Kingma, Ricardo Monti, and Aapo Hyvarinen. Variational Autoencoders and Nonlinear ICA: A Unifying Framework. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, pages 2207–2217. PMLR, June 2020.

[33] Ilyes Khemakhem, Diederik Kingma, Ricardo Monti, and Aapo Hyvärinen. Variational Autoencoders and Nonlinear ICA: A Unifying Framework. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, pages 2207–2217. PMLR, June 2020.

[34] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, January 2017.

[35] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes, December 2022.

[36] Yi-Ou Li, Tülay Adali, Wei Wang, and Vince D. Calhoun. Joint Blind Source Separation by Multiset Canonical Correlation Analysis. *IEEE Transactions on Signal Processing*, 57(10):3918–3929, October 2009.

[37] Yuening Li, Xiao Fu, and Wing-Kin Ma. Probabilistc Simplex Component Analysis Via Variational Auto-Encoding. 2024.

[38] Seppo Linnainmaa. Taylor expansion of the accumulated rounding error. *BIT Numerical Mathematics*, 16(2):146–160, June 1976.

[39] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations, June 2019.

[40] Qi Lyu and Xiao Fu. Nonlinear Multiview Analysis: Identifiability and Neural Network-Assisted Implementation. *IEEE Transactions on Signal Processing*, 68:2697–2712, 2020.

[41] Qi Lyu and Xiao Fu. Identifiability-Guaranteed Simplex-Structured Post-Nonlinear Mixture Learning via Autoencoder. *IEEE Transactions on Signal Processing*, 69:4921–4936, 2021.

[42] Qi Lyu and Xiao Fu. Provable Subspace Identification Under Post-Nonlinear Mixtures. 2022.

[43] Nestor Maslej, Loredana Fattorini, Raymond Perrault, Vanessa Parli, Anka Reuel, Erik Brynjolfsson, John Etchemendy, Katrina Ligett, Terah Lyons, James Manyika, Juan Carlos Niebles, Yoav Shoham, Russell Wald, and Jack Clark. Artificial Intelligence Index Report 2024, May 2024.

[44] Kai Wang Ng, Guo-Liang Tian, and Man-Lai Tang. *Dirichlet and Related Distributions: Theory, Methods and Applications*. Wiley Series in Probability and Statistics. Wiley, 1 edition, April 2011.

[45] E. Oja, K. Kiviluoto, and S. Malaroiu. Independent component analysis for financial time series. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, pages 111–116, Lake Louise, Alta., Canada, 2000. IEEE.

[46] John Paisley, David M Blei, and Michael I Jordan. Variational Bayesian Inference with Stochastic Search.

[47] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models, April 2022.

[48] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986.

[49] Claude Sammut. Markov Chain Monte Carlo. In Claude Sammut and Geoffrey I. Webb, editors, *Encyclopedia of Machine Learning*, pages 639–642. Springer US, Boston, MA, 2010.

[50] N.D. Sidiropoulos, R. Bro, and G.B. Giannakis. Parallel factor analysis in sensor array processing. *IEEE Transactions on Signal Processing*, 48(8):2377–2388, August 2000.

[51] Elias M. Stein and Guido Weiss. *Introduction to Fourier Analysis on Euclidean Spaces.* Number 32 in Princeton Mathematical Series. Princeton University Press, Princeton, N.J, 1975.

[52] Anisse Taleb and Christian Jutten. Source separation in post-nonlinear mixtures. *Signal Processing, IEEE Transactions on*, 47:2807–2820, November 1999.

[53] Tsung-Han Chan, Chong-Yung Chi, Yu-Min Huang, and Wing-Kin Ma. A Convex Analysis-Based Minimum-Volume Enclosing Simplex Algorithm for Hyperspectral Unmixing. *IEEE Transactions on Signal Processing*, 57(11):4418–4432, November 2009.

[54] Ruiyuan Wu, Wing-Kin Ma, Yuening Li, Anthony Man-Cho So, and Nicholas D. Sidiropoulos. Probabilistic Simplex Component Analysis. *IEEE Transactions on Signal Processing*, 70:582–599, 2022.

[55] Bo Yang, Xiao Fu, Nicholas D. Sidiropoulos, and Kejun Huang. Learning Nonlinear Mixtures: Identifiability and Algorithm. *IEEE Transactions on Signal Processing*, 68:2857–2869, 2020.

[56] Guoxu Zhou, Shengli Xie, Zuyuan Yang, Jun-Mei Yang, and Zhaoshui He. Minimum-Volume-Constrained Nonnegative Matrix Factorization: Enhanced Ability of Learning Parts. *IEEE Transactions on Neural Networks*, 22(10):1626–1637, October 2011.

[57] Michael Zibulevsky and Barak A. Pearlmutter. Blind Source Separation by Sparse Decomposition in a Signal Dictionary. *Neural Computation*, 13(4):863–882, April 2001.

[58] A. Ziehe, K. R. Müller, G. Nolte, B. M. Mackert, and G. Curio. Artifact reduction in magnetoneurography based on time-delayed second-order correlations. *IEEE transactions on bio-medical engineering*, 47(1):75–87, January 2000.