DISENTANGLED REPRESENTATIONS USING TRAINED MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose a novel method to learn disentangled representations. The ability to compute a disentangled representation is useful for many tasks because it contains information about samples from a dataset in an interpretable and compact structure. Thus development of a method that learns disentangled representations is an active area of research. In contrast to previously proposed methods, we neither require access to the values of the interpretable factors, nor to information about groups of data samples which share the values of some interpretable factors. Our proposed algorithm uses only a set of models which already have been trained on the data. With the help of the implicit function theorem we show how, using a diverse set of models that have already been trained on the data, to select a pair of data points that have a common value of interpretable factors. We prove that such an auxiliary sampler is sufficient to obtain a disentangled representation. Based on this theoretical result, we propose a loss function that the method should optimize to compute the disentangled representation. Our approach is easy to implement and shows promising results in simulations.

1 INTRODUCTION

The main goal of this paper is to determine a computational method that disentangles interpretable factors of data without access to labels. In our approach, interpretable factors are functionally independent factors that contain all the information necessary for downstream tasks of interest. For example, for a dataset containing photographs of animals, the set of interpretable factors can be the type of the nose of the animal, eyes color, and so on. We note that the set of data-generating factors may be larger than the set of interpretable factors. The background in the photos is not an interpretable factor if all the tasks of interest are related to the classification and identification of animals. While there is no standardized definition of disentangled representations, the key intuition is that a disentangled representation should capture and separate the interpretable factors of the data (Bengio, 2012; Higgins et al., 2018; Liu et al., 2021). In this paper, we assume that in a disentangled representation, each latent dimension is a diffeomorphic function of one of the interpretable factors.

The ability to compute a disentangled representation is useful for many tasks because it contains information about samples from a dataset in an interpretable and compact structure (Bengio et al., 2013; Higgins et al., 2018). Interpretability of the representation helps in tasks where users interact with a system, as they understand how it works and can provide informative feedback. Moreover, learning a disentangled representation helps for tasks where state-of-the-art machine learning-based approaches still struggle but where humans excel (Bengio, 2012). Such scenarios include learning with knowledge transfer (Tommasi et al., 2010; Huang & Wang, 2013; Pan et al., 2010), zero-shot inference (Lampert et al., 2009; Romera-Paredes & Torr, 2015) and supervised learning (Szegedy et al., 2014; Nguyen et al., 2015). A possible reason why people successfully solve these tasks is that they have a mental model that captures interpretable factors about the world. Algorithms that can capture the same explanatory factors about the world as humans will have the same ability to generalize.

The development of an algorithm that learns disentangled representations has recently become an active area of research (Detlefsen & Hauberg, 2019; Dezfouli et al., 2019; Lorenz et al., 2019; Liu et al., 2021). Since learning disentangled representations is possible only with inductive biases (Locatello et al., 2019; Hyvärinen & Pajunen, 1999), methods that learn disentangled representations use some

form of supervision (Bouchacourt et al., 2018; Fumero et al., 2021; Locatello et al., 2020a). The type of supervision ranges from full supervision (Locatello et al., 2020a; Shu et al., 2020), semi-supervision (Locatello et al., 2020b) to weak supervision (Kazemi et al., 2019; Dubrovina et al., 2019). These methods require either to have access to subsets of data that share the values of a group of interpretable factors (Bouchacourt et al., 2018; Fumero et al., 2021; Locatello et al., 2020a) or to have access to data samples with labels. (Locatello et al., 2020b).

In contrast to previous work, we neither require access to the values of the interpretable factors, nor to groups of data samples that share common values of the interpretable factors. We obtain the groups of data space that share common values of the interpretable factors using a set of models. More precisely, giving a data sample $\mathbf{x}^1$, we use the implicit function theorem to compute the intersection of preimages of the trained models. From the obtained intersection we select another data sample $\mathbf{x}^2$ that shares the values of the interpretable factors with $\mathbf{x}^1$. We prove that an auxiliary sampler selecting such pairs of points is sufficient to compute a disentangled representation. Based on our theory, we propose a loss function, minimizing which we can compute a data representation in which each dimension in the learned representation is diffiomorphic to one of the interpretable factors. Our approach is easy to implement and shows good results in simulation.

This paper is organized as follows. In Section 2, we give a definition of disentangled representations and compare it with the commonly accepted characteristics of disentangled representations. Then we summarize the properties of the trained models that our algorithm receives as input in the Section 3. In Section 4, we formally define auxiliary samplers and illustrate the theory with an example. We describe how to compute the samplers in the Section 5. Finally, we finish the paper with a summary and discussion.

2 DISENTANGLED REPRESENTATION

2.1 DEFINITION OF INTERPRETABLE FACTORS AND DISENTANGLED REPRESENTATION

At the basis of our work is the assumption that high dimensional data $D \subset \mathbb{R}^n$ has $m < n$ functionally independent factors that describe all important characteristics of it. Intuitively, this assumption means that all downstream tasks of our interest can be described as functions of m unobservable variables. There may be other data-generating factors, but they are not informative for downstream tasks. For example, in a dataset containing photographs of people, the background in the photographs is irrelevant for all face identification tasks, but the background is still a data-generating factor. In the following sentence, we formally introduce the concept of interpretable factors.

Definition 1 (interpretable factors of data)**.** *Unobserved factors* $\mathbf{z}$ *are interpretable factors of data, if they satisfy 3 conditions:*

1. *There is a map* Z*, such that* $\mathbf{z} \coloneqq (z_1(\mathbf{x}), \cdots, z_m(\mathbf{x})) \coloneqq Z(x_1, \cdots, x_n)$*;*
2. $\mathbf{z}$ *are* functionally *independent of each other, i.e., the rank of the Jacobian matrix* $\frac{\partial z_i}{\partial x_j}$ *is equal to* m *in each point of the data space;*
3. *all models of interest on the data can be expressed as functions of* $\mathbf{z}$*.*

We note, that interpretable factors are often considered statistically independent. Although the concepts of statistical and functional independence are related, they are not identical. A summary is provided in Appendix B.

We aim to construct a disentangled representation of data, which we define as follows:

Definition 2 (disentangled representation)**.** *A disentangled representation of data with respect to interpretable factors* $\mathbf{z}$ *is a map* $G : \mathbb{R}^n \to \mathbb{R}^m$ *such that*

$$G(\mathbf{x}) = (g_1(\mathbf{x}), \cdots, g_m(\mathbf{x})), \tag{1}$$

where $g_i : \mathbb{R}^n \to \mathbb{R}$ *equals to* z_i *up to diffeomorphic functions* η_i *(i.e.* η_i *are smooth and invertible):*

$$g_i(\mathbf{x}) = \eta_i(z_i(\mathbf{x})),\ \eta_i : \mathbb{R} \to \mathbb{R}. \tag{2}$$

Note that according to this definition, disentangled representations are aligned with the interpretable factors, i.e., by permuting factors from the disentangled representation, we obtain an entangled

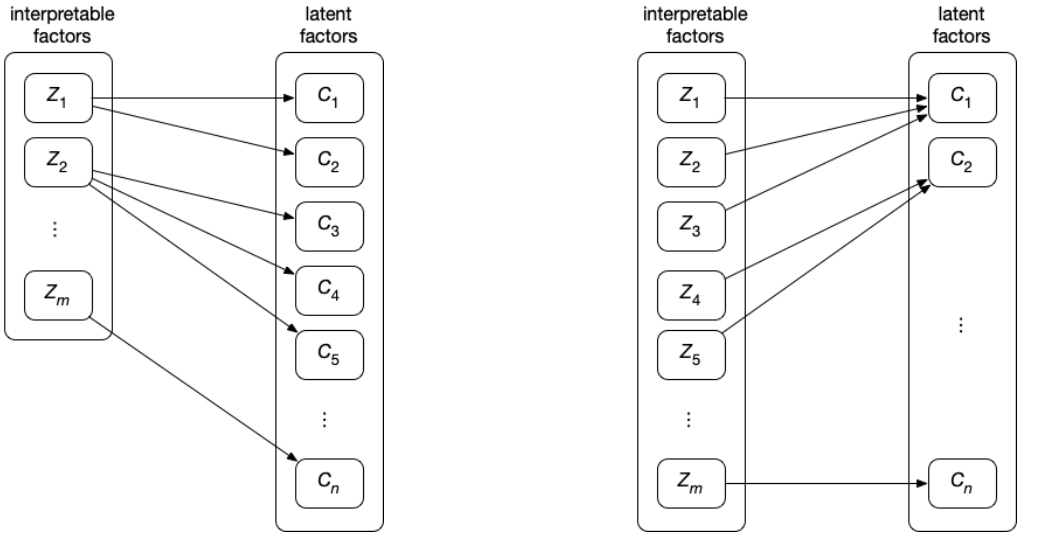
representation. This property of disentangled representations can be useful to create a machine learning model that is fair or non-discriminatory (Binns, 2018). For example, we can create an algorithm that excludes gender discrimination by having a dataset containing "gender" as the first interpretable factor. The first latent factor in a disentangled representation for such a dataset reflects gender. Thus, an algorithm that is invariant to changes in the first latent factor excludes discrimination based on gender. However an algorithm that is invariant to changes in the second latent factor does not necessarily excludes discrimination based on gender. Therefore, for this example, the alignment between latent factors and the interpretable factors is important. To achieve this alignment, we provide to the proposed algorithm a set of models for which it is known on which interpretable factors they functionally depend (see Section 3).

2.2 Proposed definition of disentanglement and commonly accepted characteristics of disentangled representations

Commonly it is accepted that disentangled representations should satisfy 2 characteristics:

Characteristic 1. *(Higgins et al., 2017; Bengio, 2012; Kim & Mnih, 2018) In a disentangled representation single latent dimensions are sensitive to changes in single interpretable factors, while being relatively invariant to changes in other factors (see Figure 1a).*

Characteristic 2. *(Kumar et al., 2018) In a disentangled representation a change in a single interpetable factor causes a highly sparse change in the representation. (see Figure 1b).This property of representations is also called* completeness *(Eastwood & Williams, 2018).*



(a) First characteristic of disentanglement.

(b) Second characteristic of disentanglement.

Figure 1: Different characteristics of disentanglement.

Representations satisfying Definition 2 satisfy Characteristics 1. To see this, let us look at 2 data samples $\mathbf{x}^1,\ \mathbf{x}^2$, which differ by the value of one of the factors in the disentangled representation, i.e.: $G(\mathbf{x}^1) = (s, g_2, \cdots, g_m)$ and $G(\mathbf{x}^2) = (s', g_2, \cdots, g_m)$, where $s \neq s'$. As each g_i is an invertible function of z_i by definition of diffeomorphic functions, it follows that $z_i(\mathbf{x}^1) = z_i(\mathbf{x}^2), \forall i \neq 1$ and $z_1(\mathbf{x}^1) \neq z_1(\mathbf{x}^2)$. That shows that a change in a single factor in the disentangled representation corresponds to a change in a single interpretale factor, which is Characteristic 1.

Representations satisfying Definition 2 also satisfy Characteristics 2. To see this, let us look at 2 data samples $\mathbf{x}^1,\ \mathbf{x}^2$, which differ by the value of one of the interpretable factors, i.e.: $Z(\mathbf{x}^1) = (s, z_2, \cdots, z_m)$ and $Z(\mathbf{x}^2) = (s', z_2, \cdots, z_m)$, where $s \neq s'$. As each of g_i is an invertible function of z_i by definition of diffeomorphic functions, it follows that $g_i(\mathbf{x}^1) = g_i(\mathbf{x}^2), \forall i \neq 1$ and $g_1(\mathbf{x}^1) \neq g_1(\mathbf{x}^2)$. That shows that a change in a single interpretable factor leads to a change in a single factor in the disentangled representation, which corresponds to Characteristic 2.

2.3 CAN WE CONSTRUCT DISENTANGLED REPRESENTATION?

Without any additional information, our goal of constructing disentangled representation is not reachable: there are several sets of interpretable factors for which the observable factors are the same. Therefore, no algorithm can distinguish between different sets of interpretable factors based on observations only and, as a consequence, create a disentangled representation corresponding only to the chosen interpretable factors. Thus, to construct disentangled representations we make additional assumptions, which are described in Section 3.

We give an example of different sets of interpretable factors for which the observable factors are the same. Let $\mathbf{z}$ are interpretable factors, Φ is a diffeomorphic deformation of $\mathbf{z}$: $\Phi : \mathbb{R}^m \to \mathbb{R}^m$, and another map to interpretable factors is $Z' \coloneqq Z \circ \Phi^{-1}$. Two maps Z and Z' have the same observable data, but different disentangled representations.

3 ASSUMPTIONS

In order to be able to build a disentangled representation additional information and assumptions are required (see Section 2.3), which we introduce in this section. **Without loss of generality, we describe assumptions for the reconstruction of g_1 only. The described assumptions can be extended in a similar way to construct other g_i.** In addition to the data set, the proposed method requires access to trained models, which we denote as $\tilde{F} \coloneqq \{\tilde{f}_1, \cdots, \tilde{f}_k\}$. As any model of our interest can be expressed as a function of interpretable factors, we can define $f_i : f_i(Z(\mathbf{x})) \coloneqq \tilde{f}_i(\mathbf{x}), \forall \tilde{f}_i \in \tilde{F}$. We denote as F the following map: $F : \mathbb{R}^m \to \mathbb{R}^k,\ F(\mathbf{z}) = (f_1(\mathbf{z}), \cdots, f_k(\mathbf{z}))$. As we show in Section 4.1 the preimage of F can provide information about $\mathbf{z}$. However, to be able to distinguish different points in $\mathbf{z}$, we require the map F to be locally injective, i.e., we assume that for every point $\mathbf{z}$ there exists a neighborhood $O(\mathbf{z})$ of it in which F is injective.

The assumptions described so far are not sufficient to distinguish z_1 from other generative factors. To overcome this, for each of the models $\tilde{f}_i$, our method requires information about whether $\tilde{f}_i$ is functionally dependent on z_1. This means that the trained models are divided into two subsets: $\tilde{F} = \tilde{F}^d \cup \tilde{F}^i$, where $\tilde{F}^i \coloneqq \{\tilde{f}_1, \cdots, \tilde{f}_l\}$ and $\tilde{F}^d \coloneqq \{\tilde{f}_{l+1}, \cdots, \tilde{f}_k\}$. Functions from $\tilde{F}^i$ are functionally **independent** of z_1, i.e., $\forall \tilde{f}_r \in \tilde{F}^i\ \forall \mathbf{z} : \frac{\partial f_r(\mathbf{z})}{\partial z_1} = 0$. Functions from $\tilde{F}^d$ are functionally **dependent** on z_1, i.e., $\forall \tilde{f}_r \in \tilde{F}^d\ \exists \mathbf{z} : \frac{\partial f_r(\mathbf{z})}{\partial z_1} \neq 0$. We also assume, that the map $F^i : \mathbb{R}^{m-1} \to \mathbb{R}^l,\ F^i(z_2, \cdots, z_m) \coloneqq (f_1(z_2, \cdots, z_m), \cdots, f_l(z_2, \cdots, z_m))$ is locally injective. That is, for every point $\overline{\mathbf{z}}_1 = (z_2, \cdots, z_m) \in \mathbb{R}^{m-1}$ there exists a neighborhood $O(\overline{\mathbf{z}}_1) \subset \mathbb{R}^{m-1}$ in which F^i is injective. As we show in Section 4.2.2, this assumption helps to select two data samples with the same values of z_1, but different values of $\overline{\mathbf{z}}_1$.

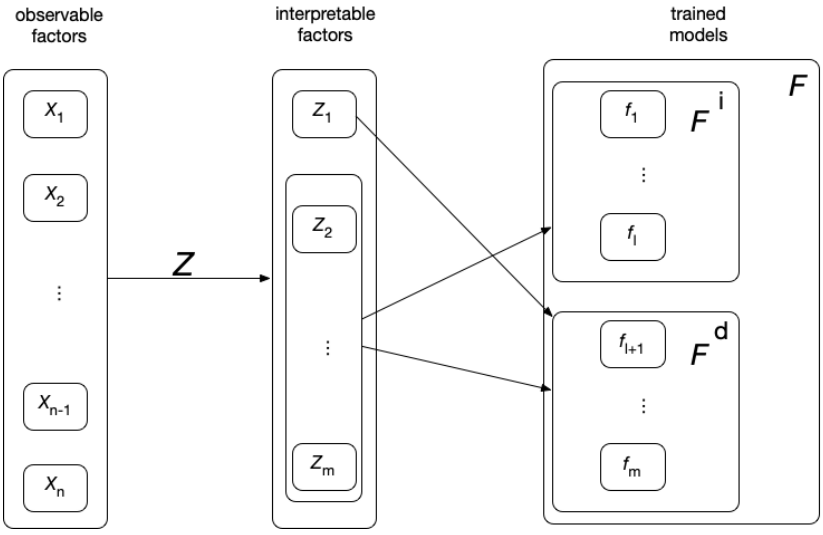


Figure 2: Summary of information we provide to algorithm and assumptions we make.

The described assumptions are summarized in Figure 2. In Appendix A, we provide a formal proof that, given the described assumptions, we can compute a disentangled representation. Having the descriptions of the assumptions, we are now ready to provide a theory of how to compute disentangled representation.

4 THEORETICAL JUSTIFICATION OF THE LOSS FUNCTION REQUIRED FOR COMPUTING DISENTANGLED REPRESENTATIONS

In this section, we first give an example of input data and show what the proposed method should compute. Then we define the loss function and give its theoretical justification.

4.1 EXAMPLE OF THE PROBLEM: INPUT DATA AND THE OUTCOME OF THE METHOD

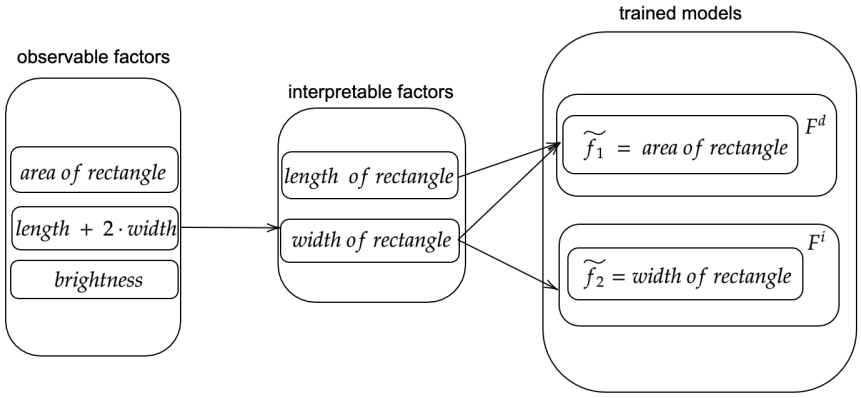


Figure 3: Example of dataset containing rectangle

As an example of input data we use a dataset containing rectangles of various shapes and brightness. Each sample from the data is described by 3 observable factors: 1.the area of the rectangle, 2.the length + 2*width, and 3.the brightness (see Figure 3). The interpretable factors are factors predefined by people. In this example, we consider the length and width of the rectangle as the interpretable factors. These are interpretable factors according to definition 1, if the tasks we are interested in do not depend on the brightness of rectangles. Examples of such problems are calculating the area of a given rectangle or calculating the length of its diagonal. In addition to a data set, we are given 2 models: $\tilde{f}_1$ and $\tilde{f}_2$. $\tilde{f}_1$ calculates the area of the rectangle, and is dependent on both interpretable factors; $\tilde{f}_2$ calculates the width of the rectangle, and is independent of the length of the rectangle (see Figure 3).

The problem that we address in this paper is to construct a disentangled representation. A disentangled latent representation of this dataset with respect to the length and width contains two latent factors. One of these factors is an invertible function of the length of the rectangles. The second factor is an invertible function of the width of the rectangles.

4.2 THEORY

4.2.1 SPURIOUS LATENT FACTORS.

The proofs below use coordinates, which we call latent factors. Specifically, the first latent factors correspond to the interpretable factors $\mathbf{z}$, while other latent factors, which we denote as $\mathbf{y} := (y_1, \cdots, y_{n-m})$, are any factors that are functionally independent of $\mathbf{z}$ (see Appendix definition 5). The latent factors $\mathbf{y}$ are not informative to downstream tasks of our interest, so we call them *spurious* latent factors. We do not require access to the latent factors, but we use the fact that for each point $\mathbf{x}^1$ there is the neighbourhood $O(\mathbf{x}^1)$, in which a diffeomorphism $\Phi_{x\to(z,y)} : O(\mathbf{x}^1) \to \mathbb{R}^n$ is defined:

$$\Phi_{x\to(z,y)}(\mathbf{x}) := (Z(\mathbf{x}), Y(\mathbf{x})) := (z_1(\mathbf{x}), \cdots, z_m(\mathbf{x}), y_1(\mathbf{x}), \cdots, y_{n-m}(\mathbf{x})). \tag{3}$$

This fact is formally proven in Appendix A.1.

To illustrate the concept of spurious latent factors we refer to the example, given in Section 4.1. For that example, the first 2 latent factors are: $1.length$ and $2.width\ of\ the\ rectangle$. The spurious latent factor could be $brightness\ of\ rectangle$, but it also could be $brightness\ of\ rectangle\ +\ area\ of\ rectangle$, since these factors functionally depend on the brightness of the rectangles.

4.2.2 THE MAIN IDEA HOW TO CONSTRUCT DISENTANGLED REPRESENTATIONS

In this section, we give a theoretical justification of the loss function required to compute a function sensitive only to changes in z_1. The described theory can be used in a similar way to compute functions sensitive to changes in other interpretable factors. To formulate the theory properly, we give notations. We denote as $\overline{\mathbf{z}}_1$ the following interpretable factors:

$$\overline{\mathbf{z}}_1 \coloneqq (z_2, \cdots, z_m). \tag{4}$$

We denote the correspondent map as $\overline{Z}_1$:

$$\overline{Z}_1 : \mathbb{R}^n \to \mathbb{R}^{m-1},\ \overline{Z}_1(\mathbf{x}) \coloneqq (z_2(\mathbf{x}), \cdots, z_m(\mathbf{x})). \tag{5}$$

Having the notation, we are ready to formulate criteria that a function is sensitive only to changes in z_1. A function is sensitive only to changes in z_1, if and only if it is invariant to changes in $\overline{\mathbf{z}}_1$ and to changes in spurious latent factors $\mathbf{y}$.

Below we propose two loss functions reflecting the described invariances: 1. a loss function reflecting invariance to changes in spurious latent factors; and 2. a loss function reflecting invariance to changes in $\overline{\mathbf{z}}_1$.

Invariance to changes in spurious latent factors To achieve invariance to $\mathbf{y}$, we show how to sample two points, witch latent factors differ only in the values of $\mathbf{y}$. To sample such pairs of points, we use level sets of the map F; and compute the closest point to $\mathbf{x}^2$ in the level set $C_F(\mathbf{x}^1) \coloneqq \{\mathbf{x} \in \mathbb{R}^n :\ F \circ Z(\mathbf{x}) = F \circ Z(\mathbf{x}^1)\}$. We denote as $M_1(\mathbf{x}^1, \mathbf{x}^2)$ the following map:

$$M_1(\mathbf{x}^1, \mathbf{x}^2) \coloneqq \underset{\mathbf{x} \in C_F(\mathbf{x}^1)}{\arg\min} \left\| \Phi_{x \to (z,y)}(\mathbf{x}^2) - \Phi_{x \to (z,y)}(\mathbf{x}) \right\|_2. \tag{6}$$

Having defined the map M_1, we give the sketch of the proof that $Z(M_1(\mathbf{x}^1, \mathbf{x}^2)) = Z(\mathbf{x}^1)$, while $Y(M_1(\mathbf{x}^1, \mathbf{x}^2)) = Y(\mathbf{x}^2)$ (the formal proof is given in Appendix in lemma 6).

Lemma 1. *Let $O(\mathbf{x}^1)$ is neighborhood of $\mathbf{x}^1$, the map F is injective in $Z(O(\mathbf{x}^1))$, and $M_1(\mathbf{x}^1, \mathbf{x}^2) \in O(\mathbf{x}^1)$, then $Z(M_1(\mathbf{x}^1, \mathbf{x}^2)) = Z(\mathbf{x}^1)$, while $Y(M_1(\mathbf{x}^1, \mathbf{x}^2)) = Y(\mathbf{x}^2)$.*

Proof. $Z(\mathbf{x}^1) \in Z(C_F(\mathbf{x}^1))$ by definition, and F is injective in $Z(O(\mathbf{x}^1))$. This means that $Z(C_F(\mathbf{x}^1) \cap O(\mathbf{x}^1)) = \{Z(\mathbf{x}^1)\}$. Consequently, any point $\mathbf{x} \in C_F(\mathbf{x}^1) \cap O(\mathbf{x}^1)$ satisfies

$$Z(\mathbf{x}) = Z(\mathbf{x}^1)$$

We proceed the proof by estimating the right side of the equation 6 and providing the point where it achieves minimum.

$$\min_{\mathbf{x}: Z(\mathbf{x}) = Z(\mathbf{x}^1)} \left\| \Phi_{x \to (z,y)}(\mathbf{x}^2) - \Phi_{x \to (z,y)}(\mathbf{x}) \right\|_2 \geq$$
$$\min_{\mathbf{x} \in \mathbb{R}^n} \left(\left\| Y(\mathbf{x}) - Y(\mathbf{x}^2) \right\|_2 + \left\| Z(\mathbf{x}^2) - Z(\mathbf{x}^1) \right\|_2 \right) \geq$$
$$\left\| Z(\mathbf{x}^2) - Z(\mathbf{x}^1) \right\|_2$$

This means that the minimum is achieved only in the points with the latent factors equal to

$$\left(Z(\mathbf{x}^1), Y(\mathbf{x}^2) \right).$$

□

Having such pairs of data points, we define the loss function $\mathcal{L}_1$, which penalizes a function for sensitivity to changes in the spurious latent variables:

$$\mathcal{L}_1(g) \coloneqq \sum_{\mathbf{x}^2 \in O(\mathbf{x}^1), \mathbf{x}^1 \in D} (g(\mathbf{x}^1) - g(M_1(\mathbf{x}^1, \mathbf{x}^2)))^2, \tag{7}$$

where D is a data set, and $O(\mathbf{x}^1) \subset \mathbb{R}^n$ is a neighborhood of $\mathbf{x}^1$. The function $g : \mathbb{R}^n \to \mathbb{R}$ is invariant to changes of spurious latent factors if and only if $\mathcal{L}_1(g) = 0$ (see formal proof in Appendix theorem 1).

To illustrate M_1 and the loss function $\mathcal{L}_1$ we use the example, given in the Section 4.1. We denote as $l(rec_1),\ w(rec_1),\ b(rec_1)$ the length, width, and brightness of the rectangle rec_1. Given two data points rec_1 and rec_2, we compute $M_1(rec_1, rec_2)$ using the level set $C_F(rec_1) = \{rec :\ l(rec) = l(rec_1),\ w(rec) = w(rec_1)\}$. The closest rectangle $rec \in C_F(rec_1)$ to rec_2 is equal to

$$\begin{aligned}
&\underset{rec \in C_F(rec_1)}{\arg\min}\ (|l(rec) - l(rec_2)|^2 + |w(rec) - w(rec_2)|^2 + |b(rec) - b(rec_2)|^2) = \\
&\underset{rec \in C_F(rec_1)}{\arg\min}\ (|l(rec_1) - l(rec_2)|^2 + |w(rec_1) - w(rec_2|^2 + |b(rec) - b(rec_2)|^2) = \\
&\qquad\qquad \underset{rec \in C_F(rec_1)}{\arg\min}\ (|b(rec) - b(rec_2)|^2).
\end{aligned}$$

Thus the closest rectangle $rec \in C_F(rec_1)$ to rec_2 has the following latent factors $(l(rec_1),\ w(rec_1),\ b(rec_2))$. Consequently, $\mathcal{L}_1$ penalises g when its values differ on a rectangle with $(l(rec_1),\ w(rec_1),\ b(rec_2))$ and on a rectangle with $(l(rec_1),\ w(rec_1),\ b(rec_1))$.

Invariance to changes in interpretable latent factors $\overline{\mathbf{z}}_1$ We achieve invariance to changes in $\overline{\mathbf{z}}_1$ in a similar way as we achieve invariance to changes in spurious latent factors. The only difference is that we sample two points, which latent factors differ only in the values of $\overline{\mathbf{z}}_1$. To sample such pairs of data points, we compute $M_2(\mathbf{x}^1, \mathbf{x}^2)$ - the closest point to $\mathbf{x}^1$ in the level set $C_{F^i}(\mathbf{x}^2) \coloneqq \{\mathbf{x} \in \mathbb{R}^n :\ F^i \circ Z(\mathbf{x}) = F^i \circ Z(\mathbf{x}^1)\}$. The latent factors of the map M_2 satisfy that $z_1(M_2(\mathbf{x}^1, \mathbf{x}^2)) = z_1(\mathbf{x}^1),\ Y(M_2(\mathbf{x}^1, \mathbf{x}^2)) = Y(\mathbf{x}^1), \overline{\mathbf{z}}_1(M_2(\mathbf{x}^1, \mathbf{x}^2)) = \overline{\mathbf{z}}_1(\mathbf{x}^2)$. The proof of this fact is analogous of the proof of Lemma 1 and is formally given in Appendix in lemma 7.

Having such pairs of data points, we define the loss function $\mathcal{L}_2$,which penalizes a function for sensitivity to changes in $\overline{\mathbf{z}}_1$:

$$\mathcal{L}_2(g) \coloneqq \sum_{\mathbf{x}^2 \in O(\mathbf{x}^1), \mathbf{x}^1 \in D} (g(\mathbf{x}^1) - g(M_2(\mathbf{x}^1, \mathbf{x}^2)))^2, \tag{8}$$

where D is a data set, and $O(\mathbf{x}^1) \subset \mathbb{R}^n$ is a neighborhood of $\mathbf{x}^1$. The function $g : \mathbb{R}^n \to \mathbb{R}$ is invariant to changes of the interpratable factors $\overline{\mathbf{z}}_1$ if and only if $\mathcal{L}_2(g) = 0$ (the formal proof is given in Appendix theorem 1).

Having the description of the two loss functions $\mathcal{L}_1$ and $\mathcal{L}_2$, we are now ready to provide a method for computing g_1.

5 Computation of the loss function

In this section, we describe the loss function required to compute a function diffeomorphic to $z_1(\mathbf{x})$. This loss function consists of 3 components:

$$\mathcal{L} \coloneqq \mathcal{L}_1 + \mathcal{L}_2 + \mathcal{L}_3. \tag{9}$$

A function g can be expressed as a function of z_1 if and only if it minimizes $\mathcal{L}_1 + \mathcal{L}_2$ (see Section 4.2). $\mathcal{L}_3$ addresses the problem that any constant function minimizes $\mathcal{L}_1 + \mathcal{L}_2$ ($\mathcal{L}_3$ is defined below in the equation 12).

The proposed method, which we denote as Aux2samp, uses 2 auxiliary samplers. The auxiliary sampler $M_1(\mathbf{x}^1, \mathbf{x}^2)$ is equal to the closest point to $\mathbf{x}^2$ in the level set $C_F(\mathbf{x}^1)$ (see Section 4.2.2). We compute $M_1(\mathbf{x}^1, \mathbf{x}^2)$ by minimizing the following expression:

$$M_1(\mathbf{x}^1, \mathbf{x}^2) = \arg\min_{\mathbf{x}} \left((1 - \mu) \cdot \left\| \tilde{F}(\mathbf{x}) - \tilde{F}(\mathbf{x}^1) \right\|_2^2 + \mu \cdot \left\| \mathbf{x} - \mathbf{x}^2 \right\|_2^2 \right), \tag{10}$$

where $\mu \in (0, 1)$ is the hyper parameter of the algorithm. The first part of the optimization functional expresses that $\mathbf{x}$ is in the level set $C_F(\mathbf{x}^1)$. The second part expresses that $\mathbf{x}$ is close to $\mathbf{x}^2$. The hyper parameter μ reflects the relative importance of these conditions. Since it is important for invariance to spurious latent factors that $z_1(M_1(\mathbf{x}^1, \mathbf{x}^2)) = z_1(\mathbf{x}^1)$ and $\mathbf{y}(M_1(\mathbf{x}^1, \mathbf{x}^2)) \neq \mathbf{y}(x^1)$, we recommend choosing μ to be less than 0.5.

The second auxiliary sampler $M_2(\mathbf{x}^1, \mathbf{x}^2)$ is equal to the closest point to $\mathbf{x}^1$ in the level set $C_{F^i}(\mathbf{x}^2)$ (see Section 4.2.2). We compute $M_2(\mathbf{x}^1, \mathbf{x}^2)$ by minimizing the following expression:

$$M_2(\mathbf{x}^1, \mathbf{x}^2) = \arg\min_{\mathbf{x}} \left((1-\lambda) \cdot \left\| \tilde{F}^i(\mathbf{x}) - \tilde{F}^i(\mathbf{x}^2) \right\|_2^2 + \lambda \cdot \left\| \mathbf{x} - \mathbf{x}^1 \right\|_2^2 \right), \tag{11}$$

where $\lambda \in (0, 1)$ is the hyper parameter of the algorithm. The first part of the optimization functional expresses that $\mathbf{x}$ is in the level set $C_F^i(\mathbf{x}^2)$. The second part expresses that $\mathbf{x}$ is close to $\mathbf{x}^1$. The hyper parameter λ reflects the relative importance of these conditions. Since it is important for invariance to latent factors $\bar{\mathbf{z}}_1$ that $z_1(M_2(\mathbf{x}^1, \mathbf{x}^2)) = z_1(\mathbf{x}^1)$ and $\bar{\mathbf{z}}_1(M_2(\mathbf{x}^1, \mathbf{x}^2)) \neq \bar{\mathbf{z}}_1(x^1)$, we recommend choosing λ to be more than 0.5 (See Appendix A.3.3 for details on the connection between the theory in Section 4.2 and the proposed optimization problems.)

The last component of the loss function, given the data set D, is defined by the following expression:

$$\mathcal{L}_3(g) \coloneqq \left(\mathcal{F}_{var}(D, g) - 1 \right)^2, \tag{12}$$

where $\mathcal{F}_{var}(D, g) \coloneqq \frac{1}{k-1} \sum_{\mathbf{x}^i \in D} \left(g(\mathbf{x}^i) - \mathcal{F}_{mean}(D, g) \right)^2$, $\mathcal{F}_{mean}(D, g) \coloneqq \frac{1}{k} \sum_{\mathbf{x}^i \in D} g(\mathbf{x}^i)$. The loss function $\mathcal{L}_3(g)$ reflects that the output of the function g is normalized, i.e. the expectation of the $g(\mathbf{x})$ is 0, while the variance of $g(\mathbf{x})$ is equal to 1. Computation of the loss function $\mathcal{L}$ is summarized in Algorithm 1.

Algorithm 1 Computation of the loss function $\mathcal{L}$

Require: Number of samples N, data set D, models $\tilde{F}$ and $\tilde{F}^i$, function $g : \mathbb{R}^n \to \mathbb{R}$

1: Randomly sample $4 \cdot N$ data points $\mathbb{D} \leftarrow \{(\mathbf{x}^{1,i}, \cdots, \mathbf{x}^{4,i}), i = 1, \cdots, N\}$

2: **for** $i \leq N$ **do**

3: $\quad \mathbf{m}_1^i \leftarrow \arg\min_{\mathbf{x}} \left((1-\mu) \cdot \left\| \tilde{F}(\mathbf{x}) - \tilde{F}(\mathbf{x}^{1,i}) \right\|_2^2 + \mu \cdot \left\| \mathbf{x} - \mathbf{x}^{2,i} \right\|_2^2 \right)$

4: $\quad \mathbf{m}_2^i \leftarrow \arg\min_{\mathbf{x}} \left((1-\lambda) \cdot \left\| \tilde{F}^i(\mathbf{x}) - \tilde{F}^i(\mathbf{x}^{3,i}) \right\|_2^2 + \lambda \cdot \left\| \mathbf{x} - \mathbf{x}^{4,i} \right\|_2^2 \right)$

5: **end for**

6: $\mathcal{L} \leftarrow \sum_{i=1}^N \left\| \mathbf{m}_1^i - g(\mathbf{x}^{1,i}) \right\|_2^2 + \sum_{i=1}^N \left\| \mathbf{m}_2^i - g(\mathbf{x}^{3,i}) \right\|_2^2 + \left(f_{var}(D, g) - 1 \right)^2$

6 EXPERIMENT

6.1 DATA PREPROCESSING

To minimize the loss functions $\mathcal{L}_1$ and $\mathcal{L}_2$, we must minimize the distance between the points $\left\| \mathbf{x} - \mathbf{x}^1 \right\|_2$. If one of observable features has much higher variance than others, its value will dominate the optimization process. To avoid this, we standardize the observable features to have mean 0 and variance 1:

$$x_i \rightsquigarrow \frac{x_i - \mathcal{F}_{mean}(D, x_i)}{\mathcal{F}_{var}(D, x_i)} \qquad where \qquad \mathbf{x} = (x_1, \cdots, x_n). \tag{13}$$

Moreover, to minimize the loss functions $\mathcal{L}_1$ and $\mathcal{L}_2$, we must also minimize the distance $\left\| \tilde{F}(\mathbf{x}) - \tilde{F}(\mathbf{x}^1) \right\|_2$. If one of the models has much higher variance than others, its value will dominate the optimization process. Thus, we also normalize output of the functions from $\tilde{F}$ to have mean 0 and variance 1:

$$\tilde{f} \rightsquigarrow \frac{\tilde{f} - \mathcal{F}_{mean}(D, \tilde{f})}{\mathcal{F}_{var}(D, \tilde{f})}, \ \forall \tilde{f} \in \tilde{F}. \tag{14}$$

6.2 EXPERIMENT

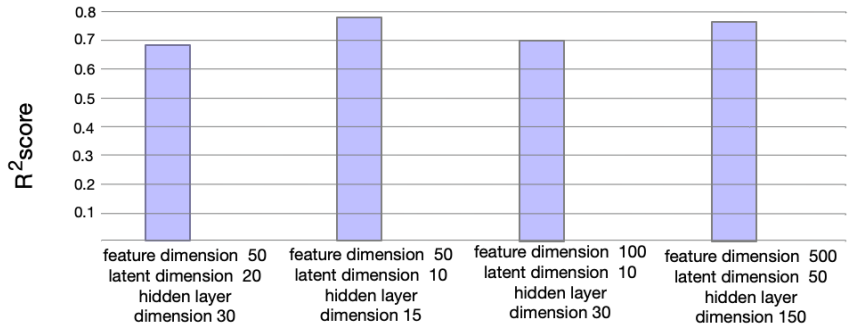


Figure 4: Result of recovering latent variable using proposed approach.

To test the performance of the proposed approach, we applied it to artificial data. In the experiments we model $\mathbf{z}$ as a result of 2 layer neural network with relu (Agarap, 2018) nonlinearity acting on the input observable variables:

$$\mathbf{z}(\mathbf{x}) = A_{z,2} \cdot relu(A_{z,1} \cdot \mathbf{x}), \tag{15}$$

where $A_{z,1}$ and $A_{z,2}$ are randomly initialized matrices. In our experiment, the functions from F^d are 2 layer neural networks with relu (Agarap, 2018) nonlinearity:

$$\tilde{f}(\mathbf{x}) = A_{f^d,2} \cdot relu(A_{f^d,1} \cdot Z(\mathbf{x})), \tag{16}$$

where $A_{f^d,1}$ and $A_{f^d,2}$ are randomly initialized matrices. The functions from F^i are 2 layer neural networks with relu (Agarap, 2018) nonlinearity:

$$\tilde{f}(\mathbf{x}) = A_{f^i,2} \cdot relu(A_{f^i,1} \cdot \overline{\mathbf{z}}_1(\mathbf{x})), \tag{17}$$

where $A_{f^i,1}$ and $A_{f^i,2}$ are randomly initialized matrices. We model g as a fully connected neural network with 2 layers and relu noninearity and find parameters of it by minimizing the loss function described in Equation 9. To measure Aux2samp performance, we use linear regressor. This linear regression is trained to predict z_1 by taking as input $g, g^2, ..., g^5$. We use the R^2 score of a trained linear model as a measure of the performance of Aux2samp.

We conduct experiments using several experimental conditions. In our experiments, we vary the dimension of the feature space, the dimension of the hidden space and the number of neurons in the hidden layer of the neural network. We repeat all experiments 10 times to obtain statistically significant results.

The proposed algorithm shows good results independently of the dimension of the feature space, the dimension of the hidden space and the number of neurons in the hidden layer of the neural network (see Figure 4). Specifically, the linear regression used to predict z_1 has R^2 score equal to $0.7 - 0.8$. Whereas linear regression, which predicts $z_i,\ i \neq 1$, using $g, g^2, \cdots, g^5$ as a feature vector, has R^2 score less than 0.001.

7 CONCLUSION

In this paper we propose a method Aux2samp to disentangle interpretable factors. Aux2samp requires a set of trained models at the input. The implicit function theorem allows us to compute preimages of the trained models and to select a pair of data points that have a common value of interpretable factors. We propose a loss function that uses these pairs of data points, and prove that by minimizing it, we obtain a disentangled representation. Based on our theory, we propose a simple recipe to compute disentangled representations in practice. In contrast with previous work the proposed method neither require labels nor information about groups of data samples which share the values of some interpretable factors. Our method is easy to implement and shows good results in simulation, independently of the dimension of features space and the number of interpretable factors.

REFERENCES

Abien Fred Agarap. Deep learning using rectified linear units (relu). *CoRR*, abs/1803.08375, 2018.

Yoshua Bengio. Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pp. 17–36. JMLR Workshop and Conference Proceedings, 2012.

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

Reuben Binns. Fairness in machine learning: Lessons from political philosophy. In *Conference on Fairness, Accountability and Transparency*, pp. 149–159. PMLR, 2018.

Diane Bouchacourt, Ryota Tomioka, and Sebastian Nowozin. Multi-level variational autoencoder: Learning disentangled representations from grouped observations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Nicki Skafte Detlefsen and Søren Hauberg. Explicit disentanglement of appearance and perspective in generative models. In *Advances in neural information processing systems*, pp. 1016–1026, 2019.

Amir Dezfouli, Hassan Ashtiani, Omar Ghattas, Richard Nock, Peter Dayan, and Cheng Soon Ong. Disentangled behavioral representations. *bioRxiv*, pp. 658252, 2019.

Anastasia Dubrovina, Fei Xia, Panos Achlioptas, Mira Shalah, Raphaël Groscot, and Leonidas J Guibas. Composite shape modeling via latent space factorization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8140–8149, 2019.

Cian Eastwood and Christopher KI Williams. A framework for the quantitative evaluation of disentangled representations. In *ICLR*, 2018.

Marco Fumero, Luca Cosmo, Simone Melzi, and Emanuele Rodolà. Learning disentangled representations via product manifold projection. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 3530–3540. PMLR, 2021.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017.

Irina Higgins, David Amos, David Pfau, Sebastien Racaniere, Loic Matthey, Danilo Rezende, and Alexander Lerchner. Towards a definition of disentangled representations. *arXiv preprint arXiv:1812.02230*, 2018.

De-An Huang and Yu-Chiang Frank Wang. Coupled dictionary and feature space learning with applications to cross-domain image synthesis and recognition. In *Proceedings of the IEEE international conference on computer vision*, pp. 2496–2503, 2013.

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

Hadi Kazemi, Seyed Mehdi Iranmanesh, and Nasser Nasrabadi. Style and content disentanglement in generative adversarial networks. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 848–856. IEEE, 2019.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2654–2663. PMLR, 2018.

Steven G Krantz and Harold R Parks. *The implicit function theorem: history, theory, and applications*. Springer Science & Business Media, 2012.

Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. Variational inference of disentangled latent concepts from unlabeled observations. In *6th International Conference on Learning Representations*, 2018.

Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *Proceedings of the IEEE Computer Vision and Pattern Recognition Conference*, pp. 951–958, 2009.

Xiao Liu, Pedro Sanchez, Spyridon Thermos, Alison Q O'Neil, and Sotirios A Tsaftaris. A tutorial on learning disentangled representations in the imaging domain. *arXiv preprint arXiv:2108.12043*, 2021.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 4114–4124. PMLR, 2019.

Francesco Locatello, Ben Poole, Gunnar Rätsch, Bernhard Schölkopf, Olivier Bachem, and Michael Tschannen. Weakly-supervised disentanglement without compromises. In *International Conference on Machine Learning*, pp. 6348–6359. PMLR, 2020a.

Francesco Locatello, Michael Tschannen, Stefan Bauer, Gunnar Rätsch, Bernhard Schölkopf, and Olivier Bachem. Disentangling factors of variation using few labels. In *8th International Conference on Learning Representations*, 2020b.

Dominik Lorenz, Leonard Bereska, Timo Milbich, and Björn Ommer. Unsupervised part-based disentangling of object shape and appearance. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 10955–10964. Computer Vision Foundation / IEEE, 2019.

Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 427–436, 2015.

Sinno Jialin Pan, Qiang Yang, et al. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

Bernardino Romera-Paredes and Philip Torr. An embarrassingly simple approach to zero-shot learning. In *International Conference on Machine Learning*, pp. 2152–2161, 2015.

Rui Shu, Yining Chen, Abhishek Kumar, Stefano Ermon, and Ben Poole. Weakly supervised disentanglement with guarantees. In *8th International Conference on Learning Representations*, 2020.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations*, 2014.

Tatiana Tommasi, Francesco Orabona, and Barbara Caputo. Safety in numbers: Learning categories from few examples with multi model knowledge transfer. In *Proceedings of the IEEE Computer Vision and Pattern Recognition Conference*, pp. 3081–3088, 2010.

A THEORY

A.1 CHOOSING LOCAL COORDINATES

Before giving the theory we make some preparation steps. First, we choose convenient local coordinates on data D. Then we choose local coordinate maps existing in L_∞-balls.

Definition 3 (L_∞-ball)**.** *L_∞-ball with radius c and the center $\mathbf{a}$ is the following set $\{\mathbf{x} \in \mathbb{R}^n : |\mathbf{x} - \mathbf{a}|_{L_\infty} < c, \mathbf{a} \in \mathbb{R}^n, c \in \mathbb{R}\}$.*

L_∞-balls possesses the property that if 2 points $\mathbf{p}^1,\ \mathbf{p}^2$ are in one L_∞-ball, then the point $\mathbf{p}^3$ for which each coordinate is equal to the corresponding coordinate of $\mathbf{p}^1$ or $\mathbf{p}^2$ is in the same L_∞-ball. We need this property of L_∞-balls in lemma 3.

Formally, we choose the coordinates in $\mathbb{R}^n$ such that the first m coordinates on D correspond to interpretable factors $\mathbf{z}$. Specifically, we choose $Y : \mathbb{R}^n \to \mathbb{R}^{n-m}$ to be a map such that the map

$$\Phi_{x\to(z,y)} : \mathbb{R}^n \to \mathbb{R}^n,\ \Phi_{x\to(z,y)}(\mathbf{x}) \coloneqq (Z(\mathbf{x}), Y(\mathbf{x})) \tag{18}$$

has rank of Jacobian $\frac{\partial \Phi_{x\to(z,y)}}{\partial x_j}$ is equal to n. Such map always exists see Lemma 4.4.7 from (Krantz & Parks, 2012).

In the following lemma we construct the local coordinate maps existing in L_∞-balls.

Lemma 2. *Lets $D \subset \mathbb{R}^n$ is a compact subset. Then we can choose a finite open cover $\cup_{i=1}^r O_i$ of $D \subset \mathbb{R}^n$, such that*

1. *$\Phi_{x\to(z,y)}$ is diffeomorphism in each of O_i*
2. *$\Phi_{x\to(z,y)}(O_i) \subset \mathbb{R}^n$ is an open L_∞-ball.*

Proof. We denote as $O_\epsilon(\mathbf{x}^1)$ epsilon neighbourhood of $\mathbf{x} \in \mathbb{R}^n$:

$$O_\epsilon(\mathbf{x}^1) \coloneqq \left\{\mathbf{x} \in \mathbb{R}^n : \left\|\mathbf{x}^1 - \mathbf{x}\right\|_2 < \epsilon\right\}. \tag{19}$$

As the rank of Jacobian of $\Phi_{x\to(z,y)}$ is equal to n, for each point $\mathbf{x} \in \mathbb{R}^n\ \exists \epsilon > 0$ such that $\Phi_{x\to(z,y)}$ is diffiomorphism in $O_\epsilon(\mathbf{x})$. Thus, by the definition of diffiomorphism, it is defined the inverse function of $\Phi_{x\to(z,y)}$ on $\Phi_{x\to(z,y)}(O_\epsilon(\mathbf{x}))$, which depends on $\mathbf{x}$:

$$\Phi^{\mathbf{x}}_{(z,y)\to x} \coloneqq \Phi^{-1}_{x\to(z,y)}|_{O_\epsilon(\mathbf{x})}, \tag{20}$$

We can choose L_∞-ball $V_\epsilon(\mathbf{x}) \subset \Phi_{x\to(z,y)}(O_\epsilon(\mathbf{x}))$, such that $\Phi_{x\to(z,y)}(\mathbf{x}) \in V_\epsilon(\mathbf{x})$. The preimage of $V_\epsilon(\mathbf{x})$ is open subset in $\mathbb{R}^n$, which we denote as

$$O(\mathbf{x}) \coloneqq \Phi^{-1}_{x\to(z,y)}(V_\epsilon(\mathbf{x})) \cap O_\epsilon(\mathbf{x}).$$

But $\cup_{\mathbf{x}\in D} O(\mathbf{x})$ is an open cover of the compact set D. Thus we can choose a finite cover of D consisting of some sets $O(\mathbf{x})$. □

Thus we construct

1. a coordinate map $\Phi_{x\to(z,y)}$
2. its inverse $\Phi^{\mathbf{x}}_{(z,y)\to x}$, which depends on $\mathbf{x}$
3. the open cover $\cup_{i=1}^r O_i$ as in the lemma 2

In what follow next we assume that these maps and the open cover are fixed.

A.2 MAIN THEOREM

Before moving further we need one more notation. We denote as $\overline{Z}_1$ the projection to $(z_2, \cdots, z_m)$, i.e.:

$$\overline{Z}_1 : \mathbb{R}^n \to \mathbb{R}^{m-1},\ \overline{Z}_1(\mathbf{x}) \coloneqq (z_2, \cdots, z_m) \tag{21}$$

In the following lemma we implicitly define 2 family of transformations M_1 and M_2. M_1 allows us to create pairs of points with the same values of $\mathbf{z}$ coordinates, but with different values of $\mathbf{y}$ coordinates. Having infinitely many such pairs of points we can design a function to be invariant to changes in $\mathbf{y}$, which is shown in the theorem 1.

M_2 allows us to create pairs of points with the same values of z_1 and $\mathbf{y}$ coordinates, but with different values of $(z_2, \cdots, z_m)$ coordinates. Having infinitely many such pairs of points we can design a function to be invariant to changes in $(z_2, \cdots, z_m)$, which is shown in the theorem 1.

Lemma 3. *There are 2 families of transformations $M_1 = \{M_{1,i} : O_i \times O_i \to O_i\}$ and $M_2 = \{M_{2,i} : O_i \times O_i \to O_i\}$ which are implicitly defined by the following properties:*

$$\Phi_{x\to(z,y)} \circ M_{1,i}(\mathbf{x}^1, \mathbf{x}^2) = (z_1(\mathbf{x}^1),\ \overline{Z}_1(\mathbf{x}^1),\ Y(\mathbf{x}^2)),\ \forall \mathbf{x}^1, \mathbf{x}^2 \in O_i. \tag{22}$$

and

$$\Phi_{x\to(z,y)} \circ M_{2,i}(\mathbf{x}^1, \mathbf{x}^2) = (z_1(\mathbf{x}^2),\ \overline{Z}_1(\mathbf{x}^1), Y(\mathbf{x}^2)),\ \forall \mathbf{x}^1, \mathbf{x}^2 \in O_i, \tag{23}$$

Proof. As we choose O_i such that $\Phi_{x\to(z,y)}(O_i)$ is L_∞-ball we can always choose points in O_i that satisfy conditions for $M_{1,i}$ and $M_{2,i}$. The image of $M_{1,i},\ M_{2,i}$ contains no more than one point from any level set of $\Phi_{x\to(z,y)}$, because $\Phi_{x\to(z,y)}$ is diffiomorphism on O_i. □

We denote as M_1 the map equals to $M_{1,i}(\mathbf{x}^1, \mathbf{x}^2)$, where $\mathbf{x}^1, \mathbf{x}^2 \in O_i$. We denote as M_2 the map equals to $M_{2,i}(\mathbf{x}^1, \mathbf{x}^2)$, where $\mathbf{x}^1, \mathbf{x}^2 \in O_i$.

Before giving our main theorem, we remind the definition of *directional derivative*

Definition 4 (Directional derivative)**.** *g is a function defined in a neighborhood of $\mathbf{x} \in \mathbb{R}^n$ and differentiable at $\mathbf{x}$. If $t : [-\epsilon, \epsilon] \to \mathbb{R}^n$ is a differentiable curve such that $t(0) = \mathbf{x}$, then the **directional derivative** of g along t can be defined as follows:*

$$\frac{\partial g}{\partial t} = \frac{d}{d\tau} g \circ t(\tau)\Big|_{\tau=0}. \tag{24}$$

In our main theorem we give the necessary and sufficient conditions the function $g : \mathbb{R}^n \to \mathbb{R}$ to be locally invariant towards the changes of z_i, y_j for $i \neq 1$, $1 \leq j \leq m - n$. Specifically, we define 2 loss functions

$$L_1(g, \mathbf{x}^1, \mathbf{x}^2) \coloneqq (g(\mathbf{x}^1) - g(M_1(\mathbf{x}^1, \mathbf{x}^2)))^2 \tag{25}$$

and

$$L_2(g, \mathbf{x}^3, \mathbf{x}^4) \coloneqq (g(\mathbf{x}^3) - g(M_2(\mathbf{x}^4, \mathbf{x}^3)))^2 \tag{26}$$

We prove that the necessary and sufficient condition for the invariance of the function to changes in $\mathbf{y}$ is the minimization of $\mathcal{L}_1$, which is defined by following equations

$$\mathcal{L}_1(g) \coloneqq \sum_{\mathbf{x}^1 \in D, \mathbf{x}^2 \in O_\epsilon(\mathbf{x}^1)} L_1(g, \mathbf{x}^1, \mathbf{x}^2) \tag{27}$$

We also prove that the necessary and sufficient condition for the invariance of the function to changes in $(z_2, \cdots, z_m)$ is the minimization of $\mathcal{L}_2$:

$$\mathcal{L}_2(g) \coloneqq \sum_{\mathbf{x}^3 \in D, \mathbf{x}^4 \in O_\epsilon(\mathbf{x}^3)} L_2(g, \mathbf{x}^3, \mathbf{x}^4) \tag{28}$$

Formally,

Theorem 1. *We denote*

$$t^{\mathbf{x}}_{z_i}(\epsilon) \coloneqq \Phi^{\mathbf{x}}_{(z,y)\to x}(\mathbf{x} + \epsilon \cdot z_i) \tag{29}$$

$$t^{\mathbf{x}}_{y_i}(\epsilon) \coloneqq \Phi^{\mathbf{x}}_{(z,y)\to x}(\mathbf{x} + \epsilon \cdot y_i) \tag{30}$$

Let g is a function invariant to changes in variables caused by M_1 or M_2, i.e. we assume:

1. *For every point $\mathbf{x}^1 \in D$ exists small number $\epsilon > 0$ such that $L_1(g, \mathbf{x}^1, \mathbf{x}^2) = 0$ for any point $\mathbf{x}^2 \in O_\epsilon(\mathbf{x}^1)$*
2. *For every point $\mathbf{x}^3 \in D$ exists small number $\epsilon > 0$ such that $L_2(g, \mathbf{x}^3, \mathbf{x}^4) = 0$ for any point $\mathbf{x}^4 \in O_\epsilon(\mathbf{x}^3)$*

Then $\frac{\partial g}{\partial t^{\mathbf{x}}_{z_i}} = 0,\ \forall \mathbf{x} \in D,\ 1 < i \leq m$.
and $\frac{\partial g}{\partial t^{\mathbf{x}}_{y_i}} = 0,\ \forall \mathbf{x} \in D,\ 0 < i \leq n - m$.

The proof of the theorem contains of 2 lemmas. The notations in lemmas are the same as in the theorem 1.

Lemma 4. *If for every point $\mathbf{x}^1 \in D$ exists small number $\epsilon > 0$ such that*

$$g(\mathbf{x}^2) = g(M_1(\mathbf{x}^2, \mathbf{x}^1))\ \forall \mathbf{x}^2 \in O_\epsilon(\mathbf{x}^1)$$

then $\frac{\partial g}{\partial t^{\mathbf{x}}_{y_i}} = 0,\ \forall \mathbf{x} \in D,\ 0 < i \leq n - m$.

Proof. Our proof is by contradiction. Lets assume that exists $\mathbf{x}^1 \in D$ such that $\frac{\partial g(\mathbf{x}^1)}{\partial t^{\mathbf{x}}_{y_i}} \neq 0$. Then $\exists \epsilon > 0 :\ g(\mathbf{x}^1) \neq g(t^{\mathbf{x}}_{y_i}(\epsilon))$. We choose ϵ to be small enough, such that $\mathbf{x}^2 \coloneqq t^{\mathbf{x}}_{y_i}(\epsilon)$ is in $O_{\epsilon_{\mathbf{p}}}(\mathbf{x}^1)$ from the condition of the lemma.

$$\Phi_{x\to(z,y)}(M_{1,i}(\mathbf{x}^2, \mathbf{x}^1)) = (Z(\mathbf{x}^2), Y(\mathbf{x}^1)) = \Phi_{x\to(z,y)}(\mathbf{x}^1).$$

$M_{1,i}(\mathbf{x}^2, \mathbf{x}^1)) = \mathbf{x}^1$, as $\Phi_{x\to(z,y)}$ diffiomorphism on O_i and $M_{1,i}(\mathbf{x}^2, \mathbf{x}^1) \in O_i$.
From assumptions of the lemma $g(\mathbf{x}^2) = g(M_{1,i}(\mathbf{x}^2, \mathbf{x}^1)) = g(\mathbf{x}^1)$. We achieve contradiction. □

Lemma 5. *If for every point $\mathbf{x}^3 \in D$ exists small number $\epsilon > 0$ such that*

$$g(\mathbf{x}^4) = g(M_2(\mathbf{x}^3, \mathbf{x}^4)),\ \forall \mathbf{x}^4 \in O_\epsilon(\mathbf{x}^3).$$

Then $\frac{\partial g}{\partial t_{z_i}} = 0,\ \forall \mathbf{x} \in D,\ 1 < i \leq m$.

Proof. Our proof is by contradiction. Lets assume that exists $\mathbf{x}^3 \in D$ such that $\frac{\partial g(\mathbf{x}^3)}{\partial t^{\mathbf{x}}_{z_i}} \neq 0$. Then $\exists \epsilon > 0 :\ g(\mathbf{p}^x_3) \neq g(t^{\mathbf{x}}_{z_i}(\epsilon))$. We choose ϵ to be small enough such that $\mathbf{x}^4 \coloneqq t^{\mathbf{x}}_{z_i}(\epsilon)$ is in $O_\epsilon(\mathbf{x}^3)$ from the second condition of the lemma.

$$\Phi_{x\to(z,y)}(M_{2,i}(\mathbf{x}^3, \mathbf{x}^4)) = (z_3(\mathbf{x}^4), P^{\overline{\mathbf{z}}_3}(\mathbf{x}^3), Y(\mathbf{x}^4)) = \Phi_{x\to(z,y)}(\mathbf{x}^3).$$

$M_{2,i}(\mathbf{x}^3, \mathbf{x}^4) = \mathbf{x}^3$ as $\Phi_{x\to(z,y)}$ diffiomorphism on O_i and $M_{2,i}(\mathbf{x}^4, \mathbf{x}^3) \in O_i$. From assumptions of the lemma $g(\mathbf{x}^4) = g(M_{2,i}(\mathbf{x}^4, \mathbf{x}^3)) = g(\mathbf{x}^3)$. We achieve contradiction.

□

Theorem 1 reduces the problem of constructing a function invariant to changes of $(z_2, \cdots, z_m)$ and $\mathbf{y}$ to an optimization problem of minimizing 2 loss functions $\mathcal{L}_1(g)$ and $\mathcal{L}_2(g)$. The last step is to build M_1 and M_2 explicitly.

A.3 CONSTRUCTING AUXILIARY MAPS

A.3.1 EXPLICIT CONSTRUCTION OF M_1

Lemma 6 reduces the problem of constructing M_1 to a simple optimisation problem. More specifically, if $C_{\tilde{F}}(\mathbf{x}^1) \subset \mathbb{R}^n$ is a level set:

$$C_{\tilde{F}}(\mathbf{x}^1) = \{\mathbf{x} \in \mathbb{R}^n : \tilde{F}(\mathbf{x}) = \tilde{F}(\mathbf{x}^1)\}$$

then $M_{1,i}(\mathbf{x}^1, \mathbf{x}^2)$ can be calculated by finding a point $\mathbf{x}$ that minimises $\left\|\Phi_{x\to(z,y)}(\mathbf{x}^2) - \Phi_{x\to(z,y)}(\mathbf{x})\right\|_2$ on the level set $C_{\tilde{F}}(\mathbf{x}^1) \cap O_i$.

Lemma 6. *Lets $F : \mathbb{R}^m \to \mathbb{R}^l$ is injective map in $Z(O_i)$. We denote as $C_F(\mathbf{x}^1) \subset \mathbb{R}^n$ the level set:*

$$C_F(\mathbf{x}^1) = \{\mathbf{x} \in \mathbb{R}^n : F \circ Z(\mathbf{x}) = F \circ Z(\mathbf{x}^1)\}. \tag{31}$$

$M_{1,i}$ defined in lemma 3 is equal to

$$M_{1,i}(\mathbf{x}^1, \mathbf{x}^2) = \underset{\mathbf{x}\in C_F(\mathbf{x}^1)\cap O_i}{\arg\min} \left\|\Phi_{x\to(z,y)}(\mathbf{x}^2) - \Phi_{x\to(z,y)}(\mathbf{x})\right\|_2. \tag{32}$$

Proof. $Z(\mathbf{x}^1) \in Z(C_F(\mathbf{x}^1))$ by definition, and F is injective in $Z(O_i)$. This means that $Z(C_F(\mathbf{x}^1) \cap O_i) = \{Z(\mathbf{x}^1)\}$. Consequently, any point $\mathbf{x} \in C_F(\mathbf{x}^1) \cap O_i$ satisfies

$$Z(\mathbf{x}) = Z(\mathbf{x}^1)$$

We proceed the proof by estimating the right side of the equation 32 and providing the point where it achieves minimum.

$$\min_{\mathbf{x}:Z(\mathbf{x})=Z(\mathbf{x}^1)} \left\|\Phi_{x\to(z,y)}(\mathbf{x}^2) - \Phi_{x\to(z,y)}(\mathbf{x})\right\|_2 \geq$$
$$\min_{\mathbf{x}\in\mathbb{R}^n} \left(\left\|Y(\mathbf{x}) - Y(\mathbf{x}^2)\right\|_2 + \left\|Z(\mathbf{x}^2) - Z(\mathbf{x}^1)\right\|_2\right) \geq$$
$$\left\|Z(\mathbf{x}^2) - Z(\mathbf{x}^1)\right\|_2$$

This means that the minimum is achieved only in the points with the coordinates

$$\left(Z(\mathbf{x}^1), Y(\mathbf{x}^2)\right).$$

We note that $\Phi_{x\to(z,y)}(O_i)$ is a L_∞-ball and $\Phi_{x\to(z,y)}$ is diffiomorphism on O_i. This involves that $\exists!\mathbf{x}^3 \in O_i$ such that $\Phi_{x\to(z,y)}(\mathbf{x}^3) = (Z(\mathbf{x}^1), Y(\mathbf{x}^2))$. □

A.3.2 EXPLICIT CONSTRUCTION OF M_2

Lemma 7 reduces the problem of constructing M_2 to a simple optimisation problem. More specifically, if $C_{\tilde{F}^i}(\mathbf{x}^1)$ is a level set:

$$C_{\tilde{F}^i}(\mathbf{x}^1) = \{\mathbf{x} \in \mathbb{R}^n : \tilde{F}^i(\mathbf{x}) = \tilde{F}^i(\mathbf{x}^1)\},$$

then $M_{2,i}(\mathbf{x}^1, \mathbf{x}^2)$ can be calculated by finding a point $\mathbf{x}$ that minimises $\left\|\Phi_{x\to(z,y)}(\mathbf{x}^2) - \Phi_{x\to(z,y)}(\mathbf{x})\right\|_2$ on the level set $C_{\tilde{F}^i}(\mathbf{x}^1) \cap O_i$.

Lemma 7. *Lets $F^i : \mathbb{R}^{m-1} \to \mathbb{R}^{k-l}$ is injective map in $\overline{Z}_1(O_i)$. We denote as $C_{F^i}(\mathbf{x}^1) \subset \mathbb{R}^n$ the level set:*

$$C_{F^i}(\mathbf{x}^1) \coloneqq \{\mathbf{x} \in \mathbb{R}^n : F^i \circ Z(\mathbf{x}) = F^i \circ Z(\mathbf{x}^1)\}. \tag{33}$$

$M_{2,i}$ defined in lemma 3 is equal to

$$M_{2,i}(\mathbf{x}^1, \mathbf{x}^2) = \underset{\mathbf{x}\in C_{F^i}(\mathbf{x}^1)\cap O_i}{\arg\min} \left\|\Phi_{x\to(z,y)}(\mathbf{x}^2) - \Phi_{x\to(z,y)}(\mathbf{x})\right\|_2. \tag{34}$$

Proof. $\overline{Z}_1(\mathbf{x}^1) \in \overline{Z}_1(C_{F^i}(\mathbf{x}^1))$ by definition and F^i is injective in $\overline{Z}_1(O_i)$. This means that $\overline{Z}_1(C_{F^i}(\mathbf{x}^1) \cap O_i) = \{\overline{Z}_1(\mathbf{x}^1)\}$. Consequently, the projection on $(z_2, \cdots, z_m)$ of any point $\mathbf{x} \in C_{F^i}(\mathbf{x}^1) \cap O_i$ coincides with the corresponding projection of $\mathbf{x}^1$:

$$\overline{Z}_1(\mathbf{x}) = \overline{Z}_1(\mathbf{x}^1)$$

Our prove consists of estimating the right side of the equation 34 and providing the point where it achieves minimum.

$$\min_{\mathbf{x}:\overline{Z}_1(\mathbf{x})=\overline{Z}_1(\mathbf{x}^1)} \left\| \Phi_{x\to(z,y)}(\mathbf{x}^2) - \Phi_{x\to(z,y)}(\mathbf{x}) \right\|_2 \geq$$

$$\min_{\mathbf{x}\in\mathbb{R}^n} \left((z_1(\mathbf{x}) - z_1(\mathbf{x}^2))^2 + \left\| Y(\mathbf{x}) - Y(\mathbf{x}^2) \right\|_2 + \left\| \overline{Z}_1(\mathbf{x}^2) - \overline{Z}_1(\mathbf{x}^1) \right\|_2 \right) \geq$$

$$\left\| \overline{Z}_1(\mathbf{x}^2) - \overline{Z}_1(\mathbf{x}^1) \right\|_2$$

This means that the minimum is achieved only in the points with the coordinates

$$\left(z_1(\mathbf{x}^2), \overline{Z}_1(\mathbf{x}^1), Y(\mathbf{x}^2) \right).$$

$\Phi_{x\to(z,y)}(O_i)$ is a L_∞-ball and $\Phi_{x\to(z,y)}$ is diffiomorphism on O_i. This involves that $\exists! \mathbf{x}^3 \in O_i$ such that $\Phi_{x\to(z,y)}(\mathbf{x}^3) = (z_1(\mathbf{x}^2), \overline{Z}_1(\mathbf{x}^1), Y(\mathbf{x}^2))$. □

A.3.3 Practical implementation of auxiliary maps

M_1 (Eq. 32) and M_2 (Eq. 34) are defined using the map $\Phi_{x\to(z,y)}$. We do not have access to $\Phi_{x\to(z,y)}$, so in our implementation we use a reasonable replacement of it. More specifically, we assume that $\Phi_{x\to(z,y)}$ is Lipschitz in D (which is a natural assumption as any function with bounded first derivative on compact set is Lipschitz). Lets the Lipschitz constant is equal to c, then

$$\left\| \Phi_{x\to(z,y)}(\mathbf{x}^2) - \Phi_{x\to(z,y)}(\mathbf{x}) \right\|_2 \leq c \cdot \left\| \mathbf{x}^2 - \mathbf{x} \right\|_2. \tag{35}$$

Thus, instead of minimising $\left\| \Phi_{x\to(z,y)}(\mathbf{x}^2) - \Phi_{x\to(z,y)}(\mathbf{x}) \right\|_2$ we minimise $\left\| \mathbf{x}^2 - \mathbf{x} \right\|_2$.

B Comparison between statistically independent and functionally independent variables.

Definition 5 (functional independence)**.** *We say that $z_1, ..., z_m$ are functionally dependent if there is an open subset $U \in \mathbb{R}^m$ such that $p((z_1, \cdots, z_n) \in U) > 0$ and a differential function $\Phi : \mathbb{R}^m \to \mathbb{R}$ such that $\Phi(z_1, ..., z_m) = 0 \; \forall \mathbf{z} \in U$. Moreover, gradient of $\Phi(\mathbf{z})$ is not equal to 0 $\forall \mathbf{z} \in U$. Otherwise $z_1, ..., z_m$ are **functionally independent**.*

Functional independence and *statistical* independence are different concepts. Indeed, variables can be statistically dependent, but functionally independent.

B.0.1 Example of statistically dependent variables which are functionally independent

For this example we introduce some notations:

- Let $a, b \in \mathbb{R}$ and $a < b$. Then we denote as $\xi \sim U[a, b]$ a random variable ξ that has uniform distribution in the segment $[a, b]$.
- We denote as $I_{\xi\in[a,b]}$ a random variable that is equal to 1 if and only if $\xi \in [a, b]$. Otherwise it equals to 0.

We give an example of 2 random variables that are statistically dependent, but functionally independent. Let $x_1 \sim U[0,1]$ is a random variables that is uniformly distributed in $[0,1]$, and let

$$x_2 \coloneqq \begin{cases} \sim U[0,0.5] & if\ x_1 \in [0,0.5] \\ \sim U[0.5,1] & if\ x_1 \in [0.5,1] \end{cases}$$

First, we assert that x_1 and x_2 are statistically dependent variables. Indeed,

$$p(x_1 < 0.5, x_2 > 0.5) = 0 \neq p(x_1 < 0.5) \cdot p(x_2 > 0.5) = 0.25. \tag{36}$$

Now we prove that x_1 and x_2 are functionally independent. Indeed, if they would be functionally dependent on the subset U with measure more than 0, then they will be dependent on some subset U_1 with measure more than 0 which lies in either $x_1 < 0.5, x_2 < 0.5$ or in $x_1 > 0.5, x_2 > 0.5$. Without loss of generality we can assume that U_1 lies in $x_1 < 0.5, x_2 < 0.5$. Than by implicit function theorem Krantz & Parks (2012) there is some function ϕ such that $z_1 = \phi(z_2)$ on some open space in $x_1 < 0.5, x_2 < 0.5$. But this contradict our assumption on distribution of x_1 and x_2.

B.0.2 Example of statistically independent variables which are functionally dependent

In this subsection we give an example of 2 random variables that are statistically independent, but functionally dependent. Assume that there are 2 independent identically distributed discrete random variables x_1, x_2

$$p(x_1 = 0) = p(x_2 = 0) = 0.5,$$
$$p(x_1 = 1) = p(x_2 = 1) = 0.5,$$
$$p(x_1 = a) = p(x_2 = b) = 0\ \forall a, b \notin \{0,1\}$$

These variables are statistically independent. But x_1, x_2 are functionally dependent. Indeed, the function

$$\Phi \coloneqq x_1 + x_2 \tag{37}$$

is equal to 0 in the interval $U \coloneqq (-0.5, 0.5)$, and

$$p(x_1, x_2 \in U) = p(x_1 = 0, x_2 = 0) = 0.25 > 0 \tag{38}$$

However, if the distribution of random variables is continuous, we cannot give such an example: when statistically independent variables have continuous distributions, then these variables are also functionally independent.

B.0.3 Connection between statistically independent random variables with continuous distribution and functional independence

In this section we formally prove that statistically independent variables with continuous distributions are also functionally independent.

Fact 1. *Let $z_1, \cdots z_m$ are random variables with the distributions F_i. Lets F_i is equal to 0 outside of some interval U_i and is continuous on U_i. Moreover,*

$$p(z_i \in U_\epsilon) > 0\ \forall open\ set\ U_\epsilon \subset U_i \tag{39}$$

Then from functional dependence of $z_1, \cdots z_m$ follows their statistical dependence.

Proof. Let $z_1, ..., z_m$ are functionally dependent. Then we can choose some open subset $U_\epsilon \subset \mathbb{R}^m$ such that $p(z_1, ..., z_m\ \in U_\epsilon) > 0$, and differentiable function F such that $F(z_1, ..., z_m) = 0$ in U_ϵ. Let $\mathbf{z}^0$ is interior point in U_ϵ and $z_1^0 \in U_1$. Without loss of generality we can assume that $\frac{\partial F}{\partial z_1}|_{\mathbf{z}^0} \neq 0$. Then by inverse function theorem Krantz & Parks (2012) there is an open set U_{ϵ_1} containing $\mathbf{z}^0$, and a continuous function f such that

$$z_1 = f(z_2, ..., z_m) \tag{40}$$

Lets $\mathbf{z}^1 \in U_{\epsilon_1}$ such that $z_1^1 \neq z_1^0$, and let $U_{\epsilon_2}(z_1^1) \subset U_1$ — is open subset which contains z_1^1 but does not contain z_1^0. As f is a continuous functions there is an open subset $V_\epsilon \subset \mathbb{R}^{m-1}$ such that $U_{\epsilon_2}(z_1^1) \cap f(V_\epsilon) = \varnothing$. Then

$$p(z_1 \in U_{\epsilon_2}(z_1^1), (z_2, ..., z_m) \in V_\epsilon) = 0$$

But

$$p(z_1 \in U_{\epsilon_2}(z_1^1)) \cdot p((z_2, ..., z_m) \in V_\epsilon) > 0.$$

Which ends the proof that $z_1, ..., z_m$ are statistically dependent.

□