Generalized Stable Weights via Neural \nGibbs Density

Yoshiaki Kitazawa

NTT DATA Mathematical Systems Inc., Data Mining Division, 
1F Shinanomachi Rengakan, 35, Shinanomachi, Shinjuku-ku, 
Tokyo, 160-0016, Japan 
kitazawa@msi.co.jp

Abstract

We present a generalized balancing method – stable weights via Neu- 
ral Gibbs Density – fully available for estimating causal effects for an 
arbitrary mixture of discrete and continuous interventions. Our weights 
are trainable through back-propagation and can be obtained with neural 
network algorithms. In addition, we also provide a method to measure 
the performance of our weights by estimating the mutual information for 
the balanced distribution. Our method is easy to implement with any 
present deep learning libraries, and the weights from it can be used in 
most state-of-art supervised algorithms.

1 Introduction

Estimating the causal effect from observational data is the central problem 
in many application domains, including public health, social sciences, clinical 
pharmacology and clinical decision making. There have been many studies 
on developing methods for estimating the causal effect for discrete or contin-
uous interventions. However, each of these methods is unsatisfactory even when 
dealing with problems only for discrete or continuous interventions. In fact, 
for discrete interventions, the following methods have been developed: The re-
weighting method by propensity score such as Inverse Propensity Weighting 
(IPW) method (Chernozhukov et al. 2018; Rosenbaum and Rubin 1984), the 
matching method (Rosenbaum and Rubin 1983; Rosenbaum and Rubin 1983) 
and the stratification method (Rosenbaum and Rubin 1983; Imbens 2004). How-
ever, these methods have difficulties to deal with causal effects for two or more 
simultaneous interventions. On the other hand, the following methods have 
developed for problems on a continuous intervention: Generalized Propen-
sity Score (GPS) (Imai and Van Dyk 2004), doubly robust estimator (Kennedy 
et al. 2017) and entropy balancing method (Hainmueller 2012; Tübbicke 2022).
As for these methods for a continuous intervention, estimators of GPS can be biased by misspecification of the parametric model for the generalized propensity score function which is generally unknown. Entropy balancing is limited in its balancing because it only removes correlations between variables of power of low degrees. We also note that entropy balancing can only handle a single intervention. In doubly robust estimator, a bias can be introduced by incorrectly specifying the structural formula that is usually unknown. Furthermore, we note that there have been few studies for methods available for estimating causal effects for an arbitrary mixture of discrete and continuous interventions.

In this paper, we give a method for estimating generalized balancing weights via neural network algorithms, with which general causal effects for an arbitrary mixture of both discrete and continuous interventions can be fully estimated. Our method is available for two or more simultaneous intervention problems, and have no limitation like entropy balancing method to estimate weights for continuous interventions. We also provide a method to measure the performance of our weights by estimating the mutual information for the balanced distribution. Our method is easy to implement with any present deep learning libraries. In addition, the weights obtained from this method can be used in most state-of-art supervised algorithms. We also perform numerical experiments on causal effect estimation with a mixture of both discrete and continuous interventions. Then, we confirm that our method performs well on the problem and even outperforms entropy balancing method for a continuous intervention problem.

This article is divided to six parts. First, we introduce the background of the study. Second, we define terminologies and concepts for causal inference. Third, we present our new method for estimation of balancing weights and also give some notes on modeling with the weights. Fourth, we report results of numerical experiments. Fifth, we discuss what our method enables. Lastly, we conclude this paper.

2 preliminaries

2.1 Notations and Definitions

Random variables are denoted by capital letters, for example, $A$. Small letters are used for values of random variables of corresponding capital letters, for example, $a$ for a value of the random variable $A$. Bold letters, $\mathbf{A}$ or $\mathbf{a}$, represent a set of variables or values of random variables. In particular, $\mathbf{V} = \{V_1, V_2, \ldots, V_n\}$ are used for observed random variables, and $\mathbf{U} = \{U_1, U_2, \ldots, U_m\}$ are used for unobserved random variables. The domain of a variable $A$, for example, is denoted by $\mathcal{X}_A$, and $\mathcal{X}_{A_1} \times \cdots \times \mathcal{X}_{A_n}$ is denoted by $\mathcal{X}_\mathbf{A}$ for $\mathbf{A} = A_1 \times \cdots \times A_n$. In addition, $\mathcal{X}_\mathbf{V}$ is written as $\Omega$. $\mathbf{V} \cup \mathbf{U}$ are supposed to be a semi-Markovian model, and $G = G_{\mathbf{V} \cup \mathbf{U}}$ denotes a causal graph for $\mathbf{V} \cup \mathbf{U}$. Each $Pa(\mathbf{A})_G$, $Ch(\mathbf{A})_G$, $An(\mathbf{A})_G$ and $De(\mathbf{A})_G$ represents the parents, children, ancestors and descen-
is defined as the probability distribution such that

\[ P(X|do(X = x)) = \sum_{V' \in \mathcal{V}' \setminus \{X \cup Y\}} \frac{P(Y, X = x, V' = v')}{P(X = x|Pa(X)_G = pa_X)}, \]  

(1)

where \( V' = V \setminus (X \cup Y) \) and \( pa_X \) represents values of \( Pa(X)_G \). The causal effect of \( X \) on \( Y \) under conditions \( Z \), denoted by \( P(Y = y|do(X = x), Z = z) \), is defined as the probability distribution such that

\[ P(Y = y|do(X = x), Z) = \frac{P(Y = y, Z|do(X = x))}{P(Z|do(X = x))}. \]  

(2)

We drop the values of intervening in and write \( do(X = x) \) simply as \( do(X) \) if not necessary in the context.

**Definition 4** (Kullback-Leibler divergence). For two probability measures \( P \) and \( Q \), the Kullback-Leibler divergence between \( P \) and \( Q \), denoted by \( D_{KL}(P||Q) \), is defined as \( D_{KL}(P||Q) := E_P[\log(\frac{dP}{dQ})] \), where \( E_P \) is the expectation for \( P \) and \( \frac{dP}{dQ} \) is the Radon-Nikodým derivative of \( P \) with respect to \( Q \).

**Definition 5** (Mutual information). For disjoint variables \( X = \{X_1, X_2, \ldots, X_n\} \subseteq \mathcal{V} \), let \( P_X \) be the joint probability measure for \( X \). For each \( i = 1, 2, \ldots, n \), \( P_{X_i} = \int_{X \setminus X_i} dP_X \) is the measure of marginal distribution of \( P_X \) for \( X_i \). The mutual information for \( X_1, X_2, \ldots, X_n \) under \( P_X \), denoted by \( MI(X_1, X_2, \ldots, X_n; P_X) \), is defined as the Kullback-Leibler divergence between \( P_X \) and \( P_{X_1} \times P_{X_2} \times \cdots \times P_{X_n} \):

\[ MI(X_1, X_2, \ldots, X_n; P_X) = D_{KL}(P_X||P_{X_1} \times P_{X_2} \times \cdots \times P_{X_n}) = E_{P_X} \left[ \log \left( \frac{dP_X}{dP_{X_1} \times dP_{X_2} \times \cdots \times dP_{X_n}} \right) \right]. \]  

(3)
3 Problem Set Up

In this paper, we consider the causal effects for joint and multi-dimensional interventions. To avoid confusion, we use different notations for each single-dimensional and multi-dimensional intervention. First, “do” is used for single-dimensional intervention and represents an operation of Pearl’s do-calculation to each dimension of the variables to intervene in:

\[ P(Y|\text{do}(X)) = P(Y|\text{do}(X_1), \text{do}(X_2), \ldots, \text{do}(X_n)), \]  

where \( X = \{X_1, X_2, \ldots, X_n\} \). Note that Eq. (4) coincides with Definition 3. Second, “do” is used for multi-dimensional intervention and represents an operation of Pearl’s do-calculation to the outside of the variables to intervene in:

\[ P(Y|\overline{\text{do}}(X_1), \overline{\text{do}}(X_2), \ldots, \overline{\text{do}}(X_n)) = P(Y|\text{do}(X)) \times P(X_1) \times P(X_2) \times \cdots \times P(X_n), \]

where \( X = \{X_1, X_2, \ldots, X_n\} \).

Now, for given disjoint sets of \( X = \{X_1, X_2, \ldots, X_n\}, Y, Z \subset V \), let

\[ P = P(Y|\overline{\text{do}}(X_1), \overline{\text{do}}(X_2), \ldots, \overline{\text{do}}(X_n), Z) \times P(Z) = P(Y|\text{do}(X), Z) \times P(X_1) \times P(X_2) \times \cdots \times P(X_n) \times P(Z). \]  

The objective of this paper is to get the stable weights which transform \( P(Y, X, Z) \) to \( \tilde{P}(Y, X, Z) \) by multiplying, or more precisely, the weights \( SW(X, Z) \) such that

\[ E_P[f(X)] = E_P[f(X) \cdot SW(X, Z)] \]

holds for any measurable function \( f \) on \( \Omega \). If we get this weights, we can estimate the Conditional Average Causal Effect (CACE) for \( P(Y|\text{do}(X_1), \text{do}(X_2), \ldots, \text{do}(X_n), Z) \), that is \( E_p[Y|X, Z] \), via state-of-art supervised machine learning algorithms with the weights given as the individual weights for each sample.

Throughout this paper, we make the two assumptions:

1. The causal effect \( P(Y|\text{do}(X)) \) is identifiable, or equivalently, \( \tilde{P} \) in (5) is identifiable. \[ \overline{\text{do}} \] symbol is useful especially when we consider interventions in a multivalued discrete variable expressed in onehot encoding, in which case we cannot express well the causal effect by \( \text{do} \) symbol. For example, consider the case for interventions in a ternary variable \( X = \{x_1, x_2, x_3\} \) and let \( X \) be expressed by \( X' = (X'_1, X'_2, X'_3) \) such that \( X'_i = 1 \) if \( X = x_i \) otherwise \( X'_i = 0 \) for \( i = 1, 2, 3 \). In this case, \( P(Y|\text{do}(X = x_i)) \) is the same as \( P(Y|\overline{\text{do}}(X' = (0, 0, 1))) \), which differs from \( P(Y|\text{do}(X' = (0, 0, 1))) \). \[ \text{The identifiability of the causal effect can be determined from the structure of the causal diagram for } P. \] One criterion for the identifiability of the causal effect is given in Shpitser and Pearl 2012. The discussion on the identifiability of the causal effect is beyond the scope of this paper.
2. Let \( P \) denote \( P(X_1, X_2, \ldots, X_n) \), that is the probability distribution of data. On the other hand, let \( Q \) denote \( P(X_1) \times P(X_2) \times \cdots \times P(X_n) \times P(Z) \), that is the balanced probability distribution of \( P \). Then, we assume that \( P \) and \( Q \) are equivalent.

4 Method

4.1 Stable Weight Method via Neural Gibbs Density

In this section, first we note that the Radon-Nikodým derivatives of a pair of equivalent probability measures can be used to change probability measure for expectation. Second, we present a new representation of the Radon-Nikodým derivatives using the energy function of gibbs density. Then, we consider how to estimate the Radon-Nikodým derivatives via a neural network algorithm. Finally, we provide a new method to estimate balancing weights generalized by Radon-Nikodým derivatives via the neural network algorithm.

Now, we note that the Radon-Nikodým derivatives of a pair of equivalent probability measures can be used to change the probability measure for expectation. For two equivalent probability measures \( P \) and \( Q \),

\[
\frac{dQ}{dP} \cdot dP = dQ \quad \text{and} \quad \frac{dP}{dQ} \cdot dQ = dP
\]

(7)

hold, or more precisely,

\[
E_P \left[ f(X) \frac{dQ}{dP} \right] = E_Q[f(X)] \quad \text{and} \quad E_Q \left[ f(X) \frac{dP}{dQ} \right] = E_P[f(X)]
\]

(8)

hold for any measurable function \( f \) on \( \Omega \). That is, \( \frac{dP}{dQ} \) transforms \( P \) to \( \tilde{P} \) in the sense that it changes the probability measure for expectation from \( P \) to \( \tilde{P} \).

Next, we present a key representation of the Radon-Nikodým derivatives for the equivalent probability measures pair \((P, Q)\). The following classical theorem suggests that the both derivatives \(\frac{dQ}{dP} \) and \(\frac{dP}{dQ} \) can be characterized by a function given from an optimization.

**Theorem 6** (Donsker-Varadhan representation). The Kullback-Leibler divergence admits the following dual representation:

\[
D_{KL}(P \parallel Q) = \sup_{T : \Omega \rightarrow \mathbb{R}} \left\{ E_P[T(X)] - \log \left( E_Q \left[ e^{T(X)} \right] \right) \right\},
\]

(9)

where the supremum is taken over all functions \( T \) such that \( E_P[T(X)] \) and \( E_Q[e^{T(X)}] \) are finite. Also the equality holds for some \( T \).

We note an important fact immediately derived from Theorem 6 as the following corollary.
Corollary 1. Let $T^*$ be the optimal function for (9), then $\frac{dQ}{dP}$ and $\frac{dP}{dQ}$ can be obtained as the gibbs density of the energy function $T^*$:

$$\frac{dQ}{dP} = \frac{1}{Z} e^{-T^*},$$

and

$$\frac{dP}{dQ} = \frac{1}{Z'} e^{T^*},$$

where $Z = E_P[e^{-T^*}]$ and $Z' = E_Q[e^{T^*}]$.

Now, we consider how to obtain $T^*$ above effectively via the neural network. With $T$ in (9) parameterized with neural network weight parameters, Belghazi et al. 2018 proposed a method to train $T$ via the neural network for the purpose of mutual information estimations. To train the function, the back-propagation by the derivative of the supremum is used: Eq. (9) is parameterized as

$$\sup_{\theta \in \Theta} \left\{ E_P[T_\theta(X)] - \log \left( E_Q \left[ e^{T_\theta(X)} \right] \right) \right\},$$

and then $T_\theta$ is trained by the gradient estimate

$$\Delta \theta = \frac{\hat{E}_P^B \left[ \nabla_\theta T_\theta(X) e^{T_\theta(X)} \right]}{\hat{E}_Q^B \left[ e^{T_\theta(X)} \right]} - \frac{\hat{E}_Q^B \left[ \nabla_\theta T_\theta(X) e^{T_\theta(X)} \right]}{\hat{E}_Q^B \left[ e^{T_\theta(X)} \right]},$$

where $\hat{E}_P^B$ and $\hat{E}_Q^B$ are the sample means of a minbatch $B$. As mentioned by Belghazi et al. 2018, the stochastic gradient estimation by Eq. (13) has a bias, since the second term of it is not linear for summation over all minibatches. In addition, we note that the common targets for this optimization, Eq. (10) and Eq. (11), are symmetric but Eq. (12) is not, which suggests the possibility for a new optimization approach different from Eq. (12).

Now, we give a new optimization method, which is also linear for summation over all minibatches.

Theorem 7. For a pair of equivalent probability measures $P$ and $Q$ which satisfy

$$E_P \left[ \left( \frac{dQ}{dP} \right)^{\frac{3}{2}} \right] < \infty \quad \text{and} \quad E_Q \left[ \left( \frac{dP}{dQ} \right)^{\frac{3}{2}} \right] < \infty,$$

let

$$T^* = \arg \sup_{T(X) : \Omega \to \mathbb{R}} E_P[e^{-T(X)}] \cdot E_Q[e^{T(X)}],$$

where the supremum is taken over all functions $T$ such that both $E_P[e^{T(X)/2}]$ and $E_Q[e^{-T(X)/2}]$ are finite. Then, (10) and (11) hold for $T^*$, and the converse is also true.
Thus, $\mathcal{L}(\theta) = -E_P[e^{-T_\theta}] \cdot E_Q[e^{T_\theta}]$ can be used as a loss function for $T_\theta$ in training neural network algorithms.

Finally, we present the main theorem which gives a new balancing method we propose in this paper.

**Theorem 8.** Given disjoint sets of $X = \{X_1, X_2, \ldots, X_n\}$, $Y, Z \subset V$ which satisfy

\begin{equation}
X = \{X_1, X_2, \ldots, X_n\} \subset \text{An}(Y)_G \text{ and } Z \cap \text{De}(X)_G \cap \text{De}(Y)_G = \phi. \quad (15)
\end{equation}

Let $P = P(X_1, X_2, \ldots, X_n)$ and $Q = P(X_1) \times P(X_2) \times \cdots \times P(X_n) \times P(Z)$. Suppose that $P$ and $Q$ are equivalent, and satisfy

\[ E_P\left[ \left( \frac{dQ}{dP} \right)^{\frac{3}{2}} \right] < \infty \quad \text{and} \quad E_Q\left[ \left( \frac{dP}{dQ} \right)^{\frac{3}{2}} \right] < \infty. \]

Let $\tilde{P} = P(Y|do(X), Z) \times P(X_1) \times P(X_2) \times \cdots \times P(X_n) \times P(Z)$.

Then, let

\begin{equation}
T^*(X_1, X_2, \ldots, X_n, Z) = \arg \sup_{T: \Omega \rightarrow \mathbb{R}} E_P[e^{-T}] \cdot E_Q[e^{T}], \quad (16)
\end{equation}

where the supremum is taken over all functions $T$ such that both $E_P[e^{T(X,Z)/2}]$ and $E_Q[e^{-T(X,Z)/2}]$ are finite. For $T^*$, it holds that

\begin{equation}
\frac{d\tilde{P}}{dP} = \frac{1}{Z} e^{-T^*(X_1, X_2, \ldots, X_n, Z)} \quad (17)
\end{equation}

and

\begin{equation}
\frac{dP}{d\tilde{P}} = \frac{1}{\tilde{Z}} e^{T^*(X_1, X_2, \ldots, X_n, Z)} \quad (18)
\end{equation}

where $Z = E_P[e^{-T^*(X_1, X_2, \ldots, X_n, Z)}]$ and $\tilde{Z} = E_Q[e^{T^*(X_1, X_2, \ldots, X_n, Z)}]$.

Here, we mention that the assumption (15) is necessary for the Eq. (17) to hold, which is derived from our Lemma 15.

To implement Theorem 8 by neural network algorithms, we parameterized Eq. (16) with neural network weight parameters and get the optimal function $T^*$ for the supremum as the optimal neural network function $T_\theta$:

\begin{equation}
T_{\theta}(X_1, X_2, \ldots, X_n, Z) = \arg \sup_{\theta \in \Theta} \hat{E}_P[e^{-T_\theta}] \cdot \hat{E}_Q[e^{T_\theta}]. \quad (19)
\end{equation}

To have the sample mean under $Q$, that is $\hat{E}_Q[e^{T_\theta}]$ in Eq. (19), a shuffling operation for samples can be used. Details on the implementation of training a NGD model are provided in Algorithm 1.
We can estimate $E_{\tilde{P}}[Y|X, Z]$, that is the CACE for $P(Y|\bar{d}o(X_1), \bar{d}o(X_2), \ldots, \bar{d}o(X_n), Z)$, by using $\frac{d\tilde{P}}{dP}$ as sample weights of a supervised algorithm, since it holds that

$$E_P \left[ f(X) \frac{d\tilde{P}}{dP} \right] = E_{\tilde{P}}[f(X)],$$  \hspace{1cm} (20)$$

for any measurable function $f$ on $\Omega$. Thus, our stable weights are the values we have from Eq. (17) with the NGD model $T_{\theta*}$ of Eq. (19). Now, we give the definition of stable weights via Neural Gibbs Density (NGD).

**Definition 9** (Stable Weights via Neural Gibbs Density). Let $T_{\theta*}$ be the optimal function for Eq. (19), then the stable weights via Neural Gibbs Density of the energy function $T_{\theta*}$, written as $SW(X_1, X_2, \ldots, X_n, Z; T_{\theta*})$, is defined as

$$SW(X_1, X_2, \ldots, X_n, Z; T_{\theta*}) = \frac{1}{Z_\theta} e^{-T_{\theta*}(X_1, X_2, \ldots, X_n, Z)},$$  \hspace{1cm} (21)$$

where $Z_\theta = \tilde{E}_P[ e^{-T_{\theta*}(X_1, X_2, \ldots, X_n, Z)} ]$.

In this paper, we distinguish the notation of $SW(\cdot)$ by the expression of the variables in the parentheses. For example, for disjoint variables $X_1, X_2, X_3 \subset V$, let $X = \{X_1, X_2\}$. Then, $SW(X, X_3; T_{\theta})$ is used to indicate the stable weights for $dP(X_1, X_2) \times dP(X_3)/dP(X_1, X_2, X_3)$. On the other hand, $SW(X_1, X_2, X_3; T_{\theta})$ denotes the stable weights for $dP(X_1) \times dP(X_2) \times dP(X_3)/dP(X_1, X_2, X_3)$. However, we drop the variables in the parentheses and write $SW(X_1, X_2, \ldots, X_n, Z; T_{\theta})$ simply as $SW_{\theta}$ if not necessary in the context.

All state-of-the-art supervised machine learning algorithms can be used with stable weights via NGD. For example, the weighted linear regression algorithm, tree-based algorithms and neural network algorithms can work with this weights. Actually, we conducted a numerical examination by using the Gradient Boosting Tree algorithm in section 5. In addition, we show the back-propagation algorithm using the stable weights for the Mean Squared Error (MSE) loss in Algorithm 3 of Appendix A.2

### 4.2 Techniques for Estimating the stable weights via NGD

We give two techniques for estimating the stable weights via Neural Gibbs Density (NGD): (i) a technique to measure the performance of the stable weights by estimating the mutual information via NGD; (ii) a technique to enhance the performance of the stable weights by using multiple NGD models.

First, we discuss how to measure the performance of our weights. Suppose we get a NGD model $T_{\theta*}$, and let $SW_{\theta*} = SW(X_1, X_2, \ldots, X_n, Z; T_{\theta*})$ be the stable weights from $T_{\theta*}$. If $SW_{\theta*}$ successfully estimates $\frac{dQ}{dP_0}$, the Kullback-Leibler divergence between $Q$ and $P_0$ will be almost nearly 0. That is, $MI(X_1, X_2, \ldots, X_n, Z; P_0)$, the mutual information under $P_0$ given by Definition 5 is very close to 0. On the
Algorithm 1 Training a Neural Gibbs Density model

Input: Data \((x_1, x_2, \ldots, x_n, z) = \{(x_1^i, x_2^i, \ldots, x_n^i, z^i) | i = 1, 2, \ldots, N\}\)

Output: An energy function \(T^*_\theta\) for Gibbs Density Stable Weights

1: \(\sigma^x_1 \leftarrow \text{SHUFFLE}(\{1 : N\})\)
2: \(\sigma^x_2 \leftarrow \text{SHUFFLE}(\{1 : N\})\)

3: \(\vdots\)
4: \(\sigma^x_n \leftarrow \text{SHUFFLE}(\{1 : N\})\)
5: \(\sigma^z \leftarrow \text{SHUFFLE}(\{1 : N\})\)
6: \(\triangleright \text{Generate shuffled indexes for } \{x_1^i, x_2^i, \ldots, x_n^i, z^i\}\)
7: \(\text{repeat}\)
8: \(\hat{E}_P \leftarrow \frac{1}{N}\sum_{i=1}^{N} e^{-T_\theta(x_1^i, x_2^i, \ldots, x_n^i, z^i)}\) \(\triangleright \text{Estimate } E_P[e^{-T_\theta}] \text{ by sample mean}\)
9: \(\hat{E}_Q \leftarrow \frac{1}{N}\sum_{i=1}^{N} e^{T_\theta(x_1^\sigma_1(i), x_2^\sigma_2(i), \ldots, x_n^\sigma_n(i), z^\sigma(i))}\) \(\triangleright \text{Estimate } E_Q[e^{T_\theta}] \text{ by sample mean}\)
10: \(\triangleright \text{Calculate Loss } L(\theta)\)
11: \(L(\theta) \leftarrow -\hat{E}_P \cdot \hat{E}_Q\)
12: \(\theta \leftarrow \theta - \nabla L(\theta)\) \(\triangleright \text{Update the parameters of } T_\theta\)
13: \(\text{until convergence}\)

contrary, if \(SW_{\theta_0}\) fails to estimate \(\frac{dQ}{dP}\), the Kullback-Leibler divergence between \(Q\) and \(P_0\) will be significantly different from 0. That is, \(MI(X_1, X_2, \ldots, X_n, Z; P_0)\) is significantly positive. This means that we can measure the performance of the stable weights by the mutual information estimation under the balanced distribution. For this purpose, we next present a new representation for the Kullback-Leibler divergence using gibbs density energy functions. Then, we show a method to estimate the mutual information using a NGD model.

The following theorem shows a new representation for the Kullback-Leibler divergence using gibbs density energy functions.

**Theorem 10.** For a pair of equivalent probability measures \(P\) and \(Q\) which satisfy

\[
E_P \left[ \left( \frac{dQ}{dP} \right)^{\frac{3}{2}} \right] < \infty \quad \text{and} \quad E_Q \left[ \left( \frac{dP}{dQ} \right)^{\frac{3}{2}} \right] < \infty,
\]

the Kullback-Leibler divergence between \(P\) and \(Q\) is given as

\[
D_{KL}(P\|Q) = -\sup_{T(\mathbf{x}):\Omega \to \mathbb{R}} \log \left( E_P[e^{-T(\mathbf{x})}] \cdot E_Q[e^{T(\mathbf{x})}] \right),
\]

where the supremum is taken over all functions \(T\) such that both \(E_P[e^{T(\mathbf{x})/2}]\) and \(E_Q[e^{-T(\mathbf{x})/2}]\) are finite. In addition, the above equality holds for \(T^*\) such that

\[
\frac{dQ}{dP} = \frac{1}{Z}e^{-T^*}
\]

(23)
and
\[ \frac{dP}{dQ} = \frac{1}{Z'} e^{T'}, \]  
where \( Z = E_P[e^{-T'(X)}] \) and \( Z' = E_Q[e^{T'(X)}] \).

Now, we present a method to estimate the mutual information via a NGD model.

**Definition 11** (Mutual Information Estimator via Neural Gibbs Density). For disjoint variables \( X_1, X_2, \ldots, X_n \subset V \), let
\[ T_{\theta_0}(X_1, X_2, \ldots, X_n) = \arg \sup_{\theta \in \Theta} \hat{E}_P[e^{-T_{\theta_0}}] \cdot \hat{E}_Q[e^{T_{\theta_0}}]. \]  
(25)

Then, the Mutual Information Estimator for \( P \) via Neural Gibbs Density is defined as
\[ \hat{MI}(X_1, X_2, \ldots, X_n; T_{\theta_0}) = -\log \left( \hat{E}_P[e^{-T_{\theta_0}}] \cdot \hat{E}_Q[e^{T_{\theta_0}}] \right). \]  
(26)

To measure the performance of stable weights from a NGD model, we estimate the mutual information for the balanced distribution from the weights. That is, in spite of the sample mean under \( P \) for Eq. (26), the sample mean under the balanced distribution is used. For example, suppose we have some weights \( SW' = \{sw^i : i = 1, 2, \ldots, N\} \), where \( sw^i \) indicates a weight for each sample \( i \) of \( N \). The balanced distribution from the weights is
\[ dP' = SW' \cdot dP. \]  
(27)

The mutual information for \( P' \) is estimated by replacing \( P \) with \( P' \) for Eq. (25) and (26) in the following way: in spite of the sample mean \( \hat{E}_P[e^{-T_{\theta_0}}] \) for these equation, we use the weighted sample mean such that
\[ \hat{E}_{P'}[e^{-T_{\theta_0}}] = \frac{1}{N} \sum_{i=1}^{N} sw^i \cdot e^{-T_{\theta_0}(x^i_1, x^i_2, \ldots, x^i_n, z^i)}. \]  
(28)

Next, we give a technique using multiple NGD models to enhance the performance of the stable weights. If we train a NGD model and the mutual information for the stable weights from the model is not sufficiently small, we can improve the weights by building an additional NGD model.

To explain in detail, suppose we get a NGD model \( T_{\theta_0} \) and let
\[ dP_0 = SW_{\theta_0} \cdot dP, \]  
(29)
where \( SW_{\theta_0} = SW(X_1, X_2, \ldots, X_n, Z; T_{\theta_0}) \) is the stable weights from \( T_{\theta_0} \).

Then, it holds that
\[ \frac{dQ}{dP} = \frac{dQ}{dP_0} \cdot \frac{dP_0}{dP} = \frac{dQ}{dP_0} \cdot SW_{\theta_0}. \]  
(30)
Therefore, even if \( SW_{\theta_0} \) fails to estimate \( dQ/dP \), the \( SW_{\theta_0} \) can be improved with \( dQ/dP_0 \). For estimating \( dQ/dP_0 \), we can use Algorithm 1 by replacing \( P \) with \( P_0 \): in spite of line 9 in Algorithm 1, estimate \( E_{P_0} \left[ e^{-T\theta} \right] \) in the same way as Eq. (28). The improved weights are obtained by using Eq. (30). More precisely, let \( T_{\theta_1} \) be a NGD model from Algorithm 1 in which line 9 is replaced with Eq. (28), then we have the improved stable weights of \( SW_{\theta_0} \) with \( SW_{\theta_1} \), which are written for \( SW(X_1, X_2, \ldots, X_n, Z; T_{\theta_0}, T_{\theta_1}) \), in the following way:

\[
SW(X_1, X_2, \ldots, X_n, Z; T_{\theta_0}, T_{\theta_1}) = \frac{SW_{\theta_0} \otimes SW_{\theta_1}}{\sum SW_{\theta_0} \otimes SW_{\theta_1}}.
\]

Here, the symbol “\( \otimes \)” is element wise product and the dominator summation is the sum over all elements.

Finally, we show a complete algorithm for building NGD models in Algorithm 2 which includes the techniques (i) and (ii). The stable weights of results from this algorithm, for example the stable weights from NGD models \( \{ T_{\theta_1}, T_{\theta_2}, \ldots, T_{\theta_n} \} \), is obtained by

\[
SW(X_1, X_2, \ldots, X_n, Z; T_{\theta_1}, T_{\theta_2}, \ldots, T_{\theta_n}) = \frac{SW_{\theta_1} \otimes SW_{\theta_2} \otimes \cdots \otimes SW_{\theta_n}}{\sum SW_{\theta_1} \otimes SW_{\theta_2} \otimes \cdots \otimes SW_{\theta_n}}.
\]

4.3 Notes for modeling the CACE with stable weights

We note that \( E_{\tilde{P}}[Y|X, Z] \) and \( E_P[Y|X, Z] \) are equal under the assumption of Theorem 8. In fact, it follows from Lemma 15 that

\[
E_{\tilde{P}}[Y|X, Z] = \int y \, d\tilde{P}(Y = y|X, Z)
= \int y \, dP(Y = y|do(X), Z)
= \int y \, dP(Y = y|X, Z)
= E_P[Y|X, Z].
\]

In practice, both estimated values for \( E_{\tilde{P}}[Y|X, Z] \) and \( E_P[Y|X, Z] \) using a machine learning algorithm are not the same. This is simply why we fail to get the true counterfactual outcomes for the probability distribution of the data, since we have to use machine learning algorithms of finite parameter and finite data in the real world.

However, this fact suggests that the usual model evaluation methods, the Hold-Out method for example, can be used to evaluate models for estimating \( E_{\tilde{P}}[Y|X, Z] \). That is, we can use the observational data for model evaluation, whenever the true counterfactual outcomes are not available. In fact, through our numerical experiments, we divided training data into two parts: training and validation
Algorithm 2 Complete Algorithm for Building Neural Gibbs Density models

Input: Data \( \{x_1, x_2, \ldots, x_n, z\} = \{(x_1, x_2', \ldots, x_n', z')|i = 1, 2, \ldots, N\} \)

Output: Energy functions \( \{T^*_\theta_i\} \) for Gibbs Densities

1: \( \mathcal{M} \leftarrow \{\} \) \( \triangleright \) Initialize \( \mathcal{M} \)
2: repeat
3: \( \sigma^\top_1 \leftarrow \) SHUFFLE\( \{1 : N\}\)
4: \( \sigma^\top_2 \leftarrow \) SHUFFLE\( \{1 : N\}\)
5: 
6: \( \sigma^\top_n \leftarrow \) SHUFFLE\( \{1 : N\}\)
7: \( \sigma^n \leftarrow \) SHUFFLE\( \{1 : N\}\)
8: \( \triangleright \) Generate shuffled indexes for \( \{x_1, x_2', \ldots, x_n', z'\} \)
9: repeat
10: \( e^{-T^\top_1} \leftarrow \) PREDICT\( ^\top_1 \)(\( x_1, x_2, \ldots, x_n, z, \mathcal{M} \))
11: \( e^{T^\top_2} \leftarrow \) PREDICT\( ^\top_2 \)(\( x_1, x_2, \ldots, x_n, z, \{\sigma^\top_1, \sigma^\top_2, \ldots, \sigma^n\}\))
12: \( \hat{E}^\top_1 \leftarrow \frac{1}{N} \sum_{x_i} e^{-T^\top_1} \)
13: \( \hat{E}^\top_2 \leftarrow \frac{1}{N} \sum_{x_i} e^{T^\top_2} \)
14: \( L(\theta) \leftarrow -\hat{E}^\top_1 \cdot \hat{E}^\top_2 \)
15: \( \theta \leftarrow \theta - \nabla L(\theta) \)
16: until convergence \( \triangleright \) Build a NGD model \( T_\theta \) under the stable weights from \( \mathcal{M} \)
17: \( \theta^\prime \leftarrow \theta \)
18: \( e^{-T_{\theta^\prime}} \leftarrow \) PREDICT\( ^{\prime} \)(\( x_1, x_2, \ldots, x_n, z, \mathcal{M} \cup \{T_{\theta^\prime}\}\))
19: \( e^{T_{\theta^\prime}} \leftarrow \) PREDICT\( ^{\prime} \)(\( x_1, x_2, \ldots, x_n, z, \{\sigma^\top_1, \sigma^\top_2, \ldots, \sigma^n\}\))
20: \( \hat{E}^{\prime} \leftarrow \frac{1}{N} \sum_{x_i} e^{-T_{\theta^\prime}} \)
21: \( \hat{E}^{\prime} \leftarrow \frac{1}{N} \sum_{x_i} e^{T_{\theta^\prime}} \)
22: \( \hat{MI}(X_1, X_2, \ldots, X_n; \mathcal{M} \cup \{\theta\}) \leftarrow -\log(\hat{E}_\theta \cdot \hat{E}^{\prime}_\theta) \)
23: \( \triangleright \) Measure the performance of the NGB stable weights from \( \mathcal{M} \cup \{T_{\theta^\prime}\}\)
24: if \( \hat{MI}(X_1, X_2, \ldots, X_n; \mathcal{M} \cup \{T_{\theta^\prime}\}) < \hat{MI}(X_1, X_2, \ldots, X_n; \mathcal{M}) \) then
25: \( \mathcal{M} \leftarrow \mathcal{M} \cup \{T_{\theta^\prime}\} \) \( \triangleright \) Update \( \mathcal{M} \) by adding \( T_{\theta^\prime} \)
26: end if
27: until \( \hat{MI}(X_1, X_2, \ldots, X_n; \mathcal{M}) = 0 \)
28: function PREDICT\( \theta \)(\( x_1, x_2, \ldots, x_n, z, \{T_{\theta_1}, T_{\theta_2}, \ldots, T_{\theta_n}\}\))
29: \( SW \leftarrow 1 \)
30: for \( T_{\theta_i} \in \{T_{\theta_1}, T_{\theta_2}, \ldots, T_{\theta_n}\} \) do
31: \( SW \leftarrow SW \otimes SW_{\theta_i} \)
32: end for
33: \( SW(\theta_1, \theta_2, \ldots, \theta_n) \leftarrow \frac{SW}{\sum_{\theta}} \)
34: \( Pred_{\theta} \leftarrow e^{-T_{\theta} \cdot \{x_1, x_2, \ldots, x_n, z\}} \otimes SW(\theta_1, \theta_2, \ldots, \theta_n) \)
35: return \( Pred_{\theta} \)
36: end function
37: function PREDICT\( \theta \)(\( x_1, x_2, \ldots, x_n, z, \{\sigma^\top_1, \sigma^\top_2, \ldots, \sigma^n\}\))
38: \( Pred_{\theta} \leftarrow \left\{ e^{T_{\theta} \cdot \{x_1, x_2', \ldots, x_n', z\}} \right\}_{i=1}^{N} \)
39: return \( Pred_{\theta} \)
40: end function
parts. Then, we trained our models by the training part, and evaluated them by the validation part to search the optimal hyper-parameters.

5 Experiments

To validate our method, we conduct two types of experiments on a synthetic data set. The data sets are created in a manner based on the way in Vegetabile et al. [2021] with some modifications for validation of our method. The experiments are performed to confirm the accuracy of estimates for the two types of causal effects: (Experiment 1.) the causal effects for a single intervention, especially for a continuous intervention; (Experiment 2.) the causal effects for a mixture of both arbitrary discrete and continuous interventions.

Data Set We use the following steps to generate the data set.

First, $W = (W_1, W_2, W_3, W_4, W_5)$ are generated from

$$(W_1, W_2, W_3, W_4) \sim \mathcal{N}(\mu, \Sigma),$$

where

$$\mu = \begin{pmatrix} -0.5 \\ 1 \\ 0 \\ 1 \end{pmatrix} \quad \text{and} \quad \Sigma = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

and

$$P(W_5 = i) = \begin{cases} 0.70 & \text{if } i = 0, \\ 0.15 & \text{if } i = 1, \\ 0.15 & \text{if } i = 2. \end{cases}$$

Then, $A$ and $Y$ are generated in the following way:

$$A \sim \mathcal{X}^2(df = 3, \mu_A),$$

where $\mu_A = \begin{cases} 5|W_1| + 6|W_2| + |W_4| & \text{if } W_5 = 0, \\ 5|W_1| + 6|W_2| + |W_4| + 1 & \text{if } W_5 = 1, \\ 5|W_1| + 6|W_2| + |W_4| + 5 & \text{if } W_5 = 2, \end{cases}$

$$Y = \frac{1}{50} \left[ (-0.15A^2 + A(W_1^2 + W_2^2) - 15) \\ + ((W_1 + 3)^2 + 2(W_2 - 25)^2 + W_3) - C + \varepsilon \right],$$

$$C = E \left[ (W_1 + 3)^2 \right] + 2E \left[ (W_2 - 25)^2 \right] + E \left[ W_3 \right],$$

$$\varepsilon \sim \mathcal{N}(0, 1).$$

Here, $\mathcal{X}^2(df = n, \mu)$ is the noncentral $\chi^2$ distribution with n degrees of freedom and a noncentral parameter $\mu$. Finally, we create new variables $X = \ldots$
\((X_1, X_2, X_3)\), as observed values of \(W\), by the following transformation:

\[
X_1 = (X_{(1,1)}, X_{(1,2)}, X_{(1,3)}), \quad \text{where} \quad X_{(1,1)} = \exp\left(\frac{W_1}{2}\right),
\]

\[
X_{(1,2)} = \frac{W_2}{1 + \exp(W_1)} + 10 \quad \text{and} \quad (37)
\]

\[
X_{(1,3)} = \frac{W_1 W_3}{25} + 0.6,
\]

\[
X_2 = (W_4 - 1)^2, \quad (38)
\]

\[
X_3 = \begin{cases} 
(1, 0) & \text{if } W_5 = 0, \\
(0, 1) & \text{if } W_5 = 1, \\
(0, 0) & \text{if } W_5 = 2. 
\end{cases} \quad (39)
\]

The above model that generates the data is similar to those developed in Vegetabile et al. 2021, but slightly different in the following points: (i) \(W_5\) is generated as a ternary discrete variable, which is binary in Vegetabile et al. 2021. (ii) The noncentral parameter of \(A\) is also changed so that it can be calculated by \(W_5\). The reason for (i) and (ii) is to validate the case involving a multivalued discrete variable which needs to be expressed as a multi-dimensional variable in onehot encoding.

**Train and Test Details** We conduct the following two experiments:

Experiment 1.

an experiment on estimating the causal effect for a single intervention, especially for a continuous intervention: \(E[Y|do(A), X]\).

Experiment 2.

an experiment on estimating the causal effect for a mixture of both arbitrary discrete and continuous interventions: \(E[Y|do(A), do(X_1), do(X_2), do(X_3)]\).

Experiment 1 is conducted to compare our method with a present method for a continuous intervention, which we mentioned in the next section. On the other hand, Experiment 2 is designed to test our method on a problem of a mixture of discrete and continuous interventions For each of experiments, we repeat the following steps 100 times where each iteration generates training and test data of size \(N = 10000\) respectively.

Step 1: We create training data with size \(N = 10000\).

Step 2: The stable weights for each experiment are estimated: We estimate \(SW(A, X : T_{\theta_n})\) for Experiment 1; and \(SW(A, X_1, X_2, X_3 : T_{\theta_n})\) for Experiment 2. The details of the neural networks and training for this step are provided in
Appendix A.3

Step 3: We create models for each experiment by using the Linear Regression (LR) or the Gradient Boosting Tree (GBT) algorithm with our weights from the previous step: models of $E[Y|do(A), X]$ for Experiment 1 is created; and models of $E[Y|do(A), \overline{do}(X_1), do(X_2), \overline{do}(X_3)]$ for Experiment 2 is created. For creating models of GBT, we use the optimal hyper-parameters we tuned.

Step 4: We create test data with size $N = 10000$ as those generated from the following distribution: the test data for Experiment 1 is generated from $P(Y|do(A), X) \times P(A) \times P(X)$, and for Experiment 2, from $P(Y|do(A), \overline{do}(X_1), do(X_2), \overline{do}(X_3)) \times P(A) \times P(X_1) \times P(X_2) \times P(X_3)$.

In order to create this data, we first generate data from the same distribution of the training data. Then, the data is shuffled by index with the following divided parts treated as a single piece of data: for Experiment 1, each $A$ and $X$ is shuffled by index; and for Experiment 2, each of $A$ and $X_1, X_2$ and $X_3$ is shuffled by index. Next, we calculate $(W_1, W_2, W_3, W_4, W_5)$ backwards from $X_1 = (X_{(1,1)}, X_{(1,2)}, X_{(1,3)}), X_2$ and $X_3$ of the data shuffled:

$W_1 = 2 \log X_{(1,1)}, \quad W_2 = X_{(1,2)} \cdot (1 + X_{(1,1)}^2),$

$W_3 = \frac{25(X_{(1,3)} - 0.6)}{2 \log X_{(1,1)}}, \quad W_4 = \sqrt{X_2 + 1},$

$W_5 = \begin{cases} 0 & \text{if } X_3 = (1, 0), \\ 1 & \text{if } X_3 = (0, 1), \\ 2 & \text{if } X_3 = (0, 0). \end{cases}$

Then, the true values of $Y$ for the causal effects are calculated by the terms of Eq. (34) without the term of $\varepsilon$.

Step 5: We estimate the average causal effects $E[Y|do(A), X]$ and $E[Y|do(A), \overline{do}(X_1), do(X_2), \overline{do}(X_3)]$ using the predictions of the models from Step 3 with the test data from Step 4. Finally, we report the Root Mean Squared Error (RMSE) between the true and the estimated values.

**Baseline Method** The main baseline method in our experiments is entropy balancing (Tübbicke 2022). We compare our method against the method for balancing $X$ with $A$ for each of moments from 1 to 4. For Experiment 1, both our method and the baseline method estimate the same target: $E[Y|do(A), X]$. However, no present methods can fully deal with the target of Experiment 2: $E[Y|do(A), \overline{do}(X_1), do(X_2), \overline{do}(X_3)]$. Therefore, the same entropy balancing as Experiment 1 is used in Experiment 2, which may be an unfair comparison for
the baseline method. In addition, we also include a “naive” estimation, estimating by the algorithms with no sample weights, as a baseline.

Results We report the average and standard errors of the RMSE between the estimated and true values of the average causal effects. The results of Experiment 1 are given in Table 1, and those of Experiment 2 in Table 2. Each result is in the form of “mean (std. err.)” from 100 simulations. Here, the results of entropy balancing with the RMSE greater than 5 are excluded, since the results of entropy balancing had the instability. As seen in Table 1, the results of NGD are significantly more accurate than those of the entropy balancing method. In addition, as seen in Table 2, our balancing method performs well on the estimation problem for a mixture of both arbitrary discrete and continuous interventions.

Table 1: Average RMSE for estimation in Experiment 1. For entropy balancing, the number to the right side of the method name, “(m)”, denotes the number of moments balanced. The results are in the form of “mean (std. err.)” from 100 simulations.

| Method               | Linear       | GBT         |
|----------------------|--------------|-------------|
| Unweighted           | 3.229(0.273) | 0.089(0.031) |
| Entropy Balancing(1) | 3.229(0.273) | 0.086(0.025) |
| Entropy Balancing(2) | 3.275(0.393) | 0.676(1.032) |
| Entropy Balancing(3) | 3.082(0.264) | 0.081(0.022) |
| Entropy Balancing(4) | 2.832(0.285) | 0.128(0.193) |
| NGD                  | **2.271(0.238)** | **0.065(0.017)** |

Table 2: Average RMSE for estimation in Experiment 2. For entropy balancing, the number to the right side of the method name, “(m)”, denotes the number of moments balanced. The results are in the form of “mean (std. err.)” from 100 simulations.

| Method               | Linear       | GBT         |
|----------------------|--------------|-------------|
| Unweighted           | 3.213(0.281) | 0.088(0.028) |
| Entropy Balancing(1) | 3.213(0.281) | 0.088(0.028) |
| Entropy Balancing(2) | 3.234(0.287) | 0.708(1.181) |
| Entropy Balancing(3) | 3.068(0.271) | 0.082(0.023) |
| Entropy Balancing(4) | 2.798(0.263) | 0.116(0.135) |
| NGD                  | **2.243(0.247)** | **0.064(0.019)** |
6 Discussion

Our method can be used for problems of an arbitrary mixture of discrete and continuous interventions. The numerical experiments have shown that our method performs well on the problem and even outperforms entropy balancing method for a continuous intervention problem. Furthermore, we note that Algorithm 1 and Algorithm 2 only require that the sample mean of input data have consistency for the population mean. Thus, this method can be used for not only \emph{i.i.d.} sampled data but also time series data. For example, data from weak stationary process has this property and is available for this method. However, we acknowledge that there is a lack of studies on the scale or structure of neural networks we need for our method.

7 Conclusion

We propose generalized balancing weights fully available for estimating causal effects for an arbitrary mixture of discrete and continuous interventions. Two methods for training our weights are provided: an optimization method to learn our weights and a method to measure the performance of the weights. This framework makes our approach more efficient and easier to obtain the balancing weights. From the numerical experiments, we show our method outperforms the entropy balancing method, and performs well in the causal effect estimation problems with a mixture of both discrete and continuous interventions.

References

Belghazi, Mohamed Ishmael et al. (2018). “Mutual information neural estimation”. In: \textit{International conference on machine learning}. PMLR, pp. 531–540.

Chernozhukov, Victor et al. (2018). \textit{Double/debiased machine learning for treatment and structural parameters}.

Hainmueller, Jens (2012). “Entropy balancing for causal effects: A multivariate reweighting method to produce balanced samples in observational studies”. In: \textit{Political analysis} 20.1, pp. 25–46.

Imai, Kosuke and David A Van Dyk (2004). “Causal inference with general treatment regimes: Generalizing the propensity score”. In: \textit{Journal of the American Statistical Association} 99.467, pp. 854–866.

Imbens, Guido W (2004). “Nonparametric estimation of average treatment effects under exogeneity: A review”. In: \textit{Review of Economics and statistics} 86.1, pp. 4–29.

Kennedy, Edward H et al. (2017). “Non-parametric methods for doubly robust estimation of continuous treatment effects”. In: \textit{Journal of the Royal Statistical Society: Series B (Statistical Methodology)} 79.4, pp. 1229–1245.

\footnote{Note that $T_\theta$ should be modeled by a recurrent neural network in the case of time series data, since it needs to learn the pattern of data from $F$ as a time series.}
Rosenbaum, Paul R and Donald B Rubin (1983). “The central role of the propensity score in observational studies for causal effects”. In: *Biometrika* 70.1, pp. 41–55.

– (1984). “Reducing bias in observational studies using subclassification on the propensity score”. In: *Journal of the American statistical Association* 79.387, pp. 516–524.

– (1985). “Constructing a control group using multivariate matched sampling methods that incorporate the propensity score”. In: *The American Statistician* 39.1, pp. 33–38.

Shpitser, Ilya and Judea Pearl (2012). “Identification of conditional interventional distributions”. In: *arXiv preprint arXiv:1206.6876*.

Tübbicke, Stefan (2022). “Entropy balancing for continuous treatments”. In: *Journal of Econometric Methods* 11.1, pp. 71–89.

Vegetabile, Brian G et al. (2021). “Nonparametric estimation of population average dose-response curves using entropy balancing weights for continuous exposures”. In: *Health Services and Outcomes Research Methodology* 21.1, pp. 69–110.

A Appendix

A.1 Proofs

In this section, let $P$ and $Q$ be a pair of equivalent probability measures, and $\mu$ be a probability measure which both $P$ and $Q$ are absolutely continuous with respect to. An example of $\mu$ is $(P + Q)/2$.

**Theorem 12** (Donsker-Varadhan representation restated). The Kullback-Leibler divergence admits the following dual representation:

$$D_{KL}(P\|Q) = \sup_{T(\mathfrak{X}):\mathfrak{X} \rightarrow \mathbb{R}} \left\{ E_P [T(X)] - \log \left( E_Q \left[ e^{T(X)} \right] \right) \right\},$$

(40)

where the supremum is taken over all functions $T$ such that $E_P[T(X)]$ and $E_Q[e^{T(X)}]$ are finite. Also the equality holds for some $T$.

**Corollary 2** (Corollary [1] restated). Let $T^*$ be the optimal function for Eq. (40), then $\frac{dQ}{dP}$ and $\frac{dP}{dQ}$ can be obtained as the gibbs density of the energy function $T^*$.

$$\frac{dQ}{dP} = \frac{1}{Z} e^{-T^*}$$

(41)

and

$$\frac{dP}{dQ} = \frac{1}{Z'} e^{T^*},$$

(42)

where $Z = E_P[e^{-T^*}]$ and $Z' = E_Q[e^{T^*}]$.

**proof of Theorem [12] and Corollary [2]**. We can see the proof, for example, as that of Theorem 4 in Belghazi et al. 2018. □
**Lemma 13.** Suppose a pair of equivalent probability measures \( P \) and \( Q \) satisfy

\[
E_P \left[ \left( \frac{dQ}{dP} \right)^{\frac{3}{2}} \right] < \infty \quad \text{and} \quad E_Q \left[ \left( \frac{dP}{dQ} \right)^{\frac{3}{2}} \right] < \infty. \quad (43)
\]

For a measurable function \( f : \Omega \to \mathbb{R} \) with \( f(x) > 0 \) (\( \mu \)-a.e. \( x \in \Omega \)) which satisfies that \( E_P[f^{-\frac{1}{2}}] < \infty \) and \( E_Q[f^{\frac{1}{2}}] < \infty \), let

\[
F[f] = \int \sqrt{f(x)} \frac{dP}{d\mu} \, d\mu + \int \frac{1}{\sqrt{f(x)}} \frac{dQ}{d\mu} \, d\mu. \quad (44)
\]

Then, \( F[f] \) is minimized if and only if \( f = \frac{dQ}{dP} \) almost every \( x \) for \( \mu \).

**Proof of Lemma 13.** First, we show \( F[f] \) is minimized if \( f_0 = \frac{dQ}{dP} \). For \( f_0 \), it holds that \( f_0(x) > 0 \) a.e. \( x \) for \( \mu \), and

\[
E_P \left[ f_0(x)^{-\frac{1}{2}} \right] = E_Q \left[ \left( \frac{dP}{dQ} \right)^{\frac{3}{2}} \right] < \infty,
\]

and

\[
E_Q \left[ f_0(x)^{\frac{1}{2}} \right] = E_P \left[ \left( \frac{dQ}{dP} \right)^{\frac{3}{2}} \right] < \infty.
\]

Thus, \( f_0 \) satisfies the assumption for the domain of \( F[f] \). Now, we note that

\[
\alpha + \beta \geq 2\sqrt{\alpha\beta}, \quad \text{for any } \alpha > 0 \text{ and } \beta > 0.
\]

(45)

holds. Then,

\[
F[f] = \int \left\{ \sqrt{f(x)} \frac{dP}{d\mu} + \frac{1}{\sqrt{f(x)}} \frac{dQ}{d\mu} \right\} \, d\mu
\]

\[
\geq \int \left\{ 2 \sqrt{\frac{dP}{d\mu}} \frac{dQ}{d\mu} \frac{1}{\sqrt{f(x)}} \frac{dQ}{d\mu} \right\} \, d\mu
\]

\[
= \int \frac{dP \, dQ}{d\mu \, d\mu} \, d\mu.
\]

On the other hand,

\[
F[f_0] = \int \sqrt{\frac{dQ}{dP}} \frac{dP}{d\mu} \, d\mu + \int \frac{dP}{dQ} \, d\mu
\]

\[
= \int \frac{dP \, dQ}{d\mu \, d\mu} \, d\mu
\]

\[
= \int \frac{dP \, dQ}{d\mu \, d\mu} \, d\mu.
\]
Therefore, we have \( F[f] \geq F[f_0] \).

Next, we show \( F[f] \) is minimized only if \( f_0 = \frac{dQ}{dP} \) a.e. for \( \mu \). Let \( \phi : \Omega \to \mathbb{R} \) be an arbitrary bounded measurable function with \( A \leq \phi(x) \leq B \) for some \( A > 0 \) and \( B > 0 \). Let

\[
I(\delta) = F[f + \delta \phi], \quad \text{for} \quad \delta \in \mathbb{R},
\]

and consider

\[
\lim_{\delta \to 0} \frac{I(\delta) - I(0)}{\delta} = \lim_{\delta \to 0} \left\{ \int \frac{1}{\delta} \left( \sqrt{f(x)} + \delta \phi(x) - \sqrt{f(x)} \right) \frac{dP}{d\mu} d\mu 
\right.
\]

\[
+ \int \frac{1}{\delta} \left( \frac{1}{\sqrt{f(x) + \delta \phi(x)}} - \frac{1}{\sqrt{f(x)}} \right) \frac{dQ}{d\mu} d\mu \right\}
\]

\[
= \lim_{\delta \to 0} \left\{ \int \frac{1}{\delta} \left( \sqrt{f(x)} + \delta \phi(x) - \sqrt{f(x)} \right) \frac{dP}{d\mu} d\mu 
\right.
\]

\[
+ \frac{1}{\delta} \left( \frac{1}{\sqrt{f(x) + \delta \phi(x)}} - \frac{1}{\sqrt{f(x)}} \right) \frac{dQ}{d\mu} d\mu \right\} d\mu.
\]

We will evaluate the upper bound for the absolute value of the integrand of (47).

Taking the absolute value for the integrand of (47), we have

\[
\left| \frac{1}{\delta} \left( \sqrt{f(x)} + \delta \phi(x) - \sqrt{f(x)} \right) \frac{dP}{d\mu} 
\right.
\]

\[
+ \frac{1}{\delta} \left( \frac{1}{\sqrt{f(x) + \delta \phi(x)}} - \frac{1}{\sqrt{f(x)}} \right) \frac{dQ}{d\mu} \right| 
\]

\[
\leq \left| \frac{1}{\delta} \left( \sqrt{f(x)} + \delta \phi(x) - \sqrt{f(x)} \right) \frac{dP}{d\mu} \right|
\]

\[
+ \frac{1}{\delta} \left( \frac{1}{\sqrt{f(x) + \delta \phi(x)}} - \frac{1}{\sqrt{f(x)}} \right) \frac{dQ}{d\mu} \right|.
\]
For evaluating the upper bound for the first term of (48), we obtain

\[
\frac{1}{\delta} \left| \sqrt{f(x) + \delta \phi(x)} - \sqrt{f(x)} \right| \\
= \frac{1}{\delta} \left| \sqrt{f(x) + \delta \phi(x)} - \sqrt{f(x)} \right| \cdot \frac{\sqrt{f(x) + \delta \phi(x)} + \sqrt{f(x)}}{\sqrt{f(x) + \delta \phi(x)} + \sqrt{f(x)}} \\
= \frac{1}{\delta} \left| \frac{f(x) + \delta \phi(x) - f(x)}{\sqrt{f(x) + \delta \phi(x)} + \sqrt{f(x)}} \right| \\
= \left| \frac{\phi(x)}{\sqrt{f(x) + \delta \phi(x)} + \sqrt{f(x)}} \right| \\
\leq \left| \frac{\phi(x)}{2 \sqrt{f(x)}} \right| \\
\leq \frac{B}{2} f(x)^{-\frac{1}{2}}. 
\]  

(49)

For evaluating the upper bound for the second term of (48), we note that

\[
\left| 1 - \frac{1}{\sqrt{f + \delta \phi}} \right| \\
\leq \left| \sqrt{f^{-1} + \delta \phi^{-1}} - 1 \right| \times \left| \sqrt{f^{-1} + \delta \phi^{-1}} - \frac{1}{\sqrt{f + \delta \phi}} \right| 
\]  

(50)

holds. This is obtained as follows: By using (45), for \( \delta > 0 \), we obtain

\[
(f^{-1} + \delta \phi^{-1})(f + \delta \phi) = 1 + \delta^2 + \delta f^{-1} \phi + \delta f \phi^{-1} \\
\geq 2 + \delta^2 + 2\delta \sqrt{f^{-1} \phi \cdot f \phi^{-1}} = (1 + \delta)^2.
\]

Thus, for \( \delta > 0 \),

\[
\frac{1}{\sqrt{f}} \leq \frac{1}{\sqrt{f + \delta \phi}} \leq \frac{\sqrt{f^{-1} + \delta \phi^{-1}}}{1 + \delta}
\]

holds. Similarly, for \( \delta < 0 \), we obtain

\[
(f^{-1} + \delta \phi^{-1})(f + \delta \phi) = 1 + \delta^2 + \delta f^{-1} \phi + \delta f \phi^{-1} \\
\leq 2 + \delta^2 + 2\delta \sqrt{f^{-1} \phi \cdot f \phi^{-1}} = (1 + \delta)^2.
\]

Thus, for \( \delta < 0 \),

\[
\frac{\sqrt{f^{-1} + \delta \phi^{-1}}}{1 + \delta} \leq \frac{1}{\sqrt{f + \delta \phi}} \leq \frac{1}{\sqrt{f}}
\]

holds.
By using (50), we have

\[
\frac{1}{\delta} \left| \frac{1}{\sqrt{f(x) + \delta \phi(x)}} - \frac{1}{\sqrt{f(x)}} \right| \\
\leq \frac{1}{\delta} \left| \frac{\sqrt{f(x)^{-1} + \delta \phi(x)^{-1}}}{(1 + \delta)} - \sqrt{f(x)^{-1}} \right| \\
= \frac{1}{\delta} \left| \frac{\sqrt{f(x)^{-1} + \delta \phi(x)^{-1}}}{(1 + \delta)} - \sqrt{f(x)^{-1}} \right| \times \frac{\sqrt{f(x)^{-1} + \delta \phi(x)^{-1} + \sqrt{f(x)^{-1}}}}{\sqrt{f(x)^{-1} + \delta \phi(x)^{-1} + \sqrt{f(x)^{-1}}}} \\
= \frac{1}{\delta} \left| \frac{f(x)^{-1} + \delta \phi(x)^{-1} - f(x)^{-1}}{(1 + \delta) \left( \sqrt{f(x)^{-1} + \delta \phi(x)^{-1} + \sqrt{f(x)^{-1}}} \right)} \right| \\
= \frac{1}{1 + \delta} \left| \frac{\phi^{-1}}{\sqrt{f(x)^{-1} + \delta \phi(x)^{-1} + \sqrt{f(x)^{-1}}}} \right| \\
\leq \frac{1}{1 + \delta} \left| \frac{A}{\sqrt{f(x)^{-1} + \sqrt{f(x)^{-1}}}} \right| \\
= \frac{A}{2(1 + \delta)} f(x)^{\frac{1}{2}}. \\
\text{ (51)}
\]

From (48), (49) and (51), it holds that

\[
\left| \frac{1}{\delta} \left( \sqrt{f(x) + \delta \phi(x)} - \sqrt{f(x)} \right) \right| \left| \frac{dP}{d\mu} \right| + \frac{1}{\delta} \left| \frac{1}{\sqrt{f(x) + \delta \phi(x)}} - \frac{1}{\sqrt{f(x)}} \right| \left| \frac{dQ}{d\mu} \right| \\
\leq \frac{B}{2} f(x)^{-\frac{1}{4}} \frac{dP}{d\mu} + \frac{A}{2(1 + \delta)} f(x)^{\frac{1}{2}} \frac{dQ}{d\mu} \\
\leq C_1 f(x)^{-\frac{1}{4}} \frac{dP}{d\mu} + C_2 f(x)^{\frac{1}{2}} \frac{dQ}{d\mu}, \\
\text{ (52)}
\]

where \( C_1 > 0 \) and \( C_2 > 0 \) are some constant values. The integral of the right side of (52) for \( \mu \) is finite because of the assumption on \( f \), so that the limitation
can be replaced with the integral for (47). Then, we have
\[
\lim_{\delta \to 0} \frac{I(\delta) - I(0)}{\delta} = \int \lim_{\delta \to 0} \frac{1}{\delta} \left( \sqrt{f(x) + \delta \phi(x)} - \sqrt{f(x)} \right) \frac{dP}{d\mu} d\mu
\]
\[
+ \int \lim_{\delta \to 0} \frac{1}{\delta} \left( \frac{1}{\sqrt{f(x) + \delta \phi(x)}} - \frac{1}{\sqrt{f(x)}} \right) \frac{dQ}{d\mu} d\mu
\]
\[
= \int \frac{\phi(x)}{2} f(x)^{-\frac{1}{2}} \frac{dP}{d\mu} + \int \frac{1}{2} \phi(x) f(x)^{-\frac{3}{2}} \frac{dQ}{d\mu} d\mu
\]
\[
= \int \phi(x) \left[ \frac{1}{2} f(x)^{-\frac{1}{2}} \frac{dP}{d\mu} - \frac{1}{2} f(x)^{-\frac{3}{2}} \frac{dQ}{d\mu} \right] d\mu. \tag{53}
\]
Now, \(I(\delta)\) is minimized when \(\delta = 0\), since \(f\) minimize \(F[f]\). Then, we see that
\[
I'(0) = \lim_{\delta \to 0} \frac{I(\delta) - I(0)}{\delta} = 0,
\]
and it follows that
\[
\int \phi(x) \left[ \frac{1}{2} f(x)^{-\frac{1}{2}} \frac{dP}{d\mu} - \frac{1}{2} f(x)^{-\frac{3}{2}} \frac{dQ}{d\mu} \right] d\mu = 0. \tag{54}
\]
Since \(\phi(x)\) is an arbitrary positive bounded function, (54) implies that
\[
\frac{1}{2} f(x)^{-\frac{1}{2}} \frac{dP}{d\mu} - \frac{1}{2} f(x)^{-\frac{3}{2}} \frac{dQ}{d\mu} = 0 \quad \text{a.e. } x \text{ for } \mu. \tag{55}
\]
Multiplying both sides of (55) by \(2f(x)^{\frac{3}{2}}\), we have
\[
f(x) \frac{dP}{d\mu} - \frac{dQ}{d\mu} = 0 \quad \text{a.e. } x \text{ for } \mu,
\]
then we obtain \(f(x) = \frac{dQ}{dP}\) a.e. \(x\) for \(\mu\).
This completes the proof. \(\square\)

**Theorem 14** (Theorem 7 restated). For a pair of equivalent probability measures \(P\) and \(Q\) which satisfy
\[
E_P \left[ \left( \frac{dQ}{dP} \right)^{\frac{3}{2}} \right] < \infty \quad \text{and} \quad E_Q \left[ \left( \frac{dP}{dQ} \right)^{\frac{3}{2}} \right] < \infty, \tag{56}
\]
let
\[
T^* = \arg \sup_{T(\cdot): \Omega \to \mathbb{R}} E_P[e^{-T(X)}] \cdot E_Q[e^{T(X)}], \tag{57}
\]
where the supremum is taken over all functions \(T\) such that both \(E_P[e^{T(X)/2}]\) and \(E_Q[e^{-T(X)/2}]\) are finite. Then, (41) and (42) hold for \(T^*\), and the converse is also true.
proof of Theorem 14. Firstly, we show (10) and (11) hold for \( T^* \). Let \( \mathcal{F} = \{ f : f = \frac{1}{Z_T} e^{-T(X)} \}, Z_T = E_P[e^{-T(X)}] \) for \( T(X) : \Omega \to \mathbb{R} \), and \( \mathcal{F}^+ = \{ f \in \mathcal{F} : E_P[f^{-\frac{1}{2}}] < \infty \text{ and } E_Q[f^{\frac{1}{2}}] < \infty \} \) and \( \mathcal{T}^+ = \{ T(X) : \Omega \to \mathbb{R} : E_P[e^{T(X)/2}] < \infty \text{ and } E_Q[e^{-T(X)/2}] < \infty \} \). Note that, let \( f_0 = \frac{dQ}{dP} \), and

\[
T_0(x) = \begin{cases} 
-\log (f_0), & \text{if } f_0 = \frac{dP}{dQ} > 0, \\
0, & \text{otherwise}, 
\end{cases} 
\tag{58}
\]

then \( \frac{dQ}{dP} = \frac{1}{Z_{T_0}} e^{-T_0(X)} \) where \( Z_{T_0} = E_P[e^{-T_0(X)}] \). In addition, it holds that

\[
E_P \left[ f_0^{-\frac{1}{2}} \right] = E_P \left[ e^{\frac{1}{2} T_0(X)} \right] = E_Q \left[ \left( \frac{dP}{dQ} \right)^{\frac{1}{2}} \right] < \infty,
\]

and

\[
E_Q \left[ f_0^{\frac{1}{2}} \right] = E_Q \left[ e^{-\frac{1}{2} T_0(X)} \right] = E_P \left[ \left( \frac{dQ}{dP} \right)^{\frac{1}{2}} \right] < \infty.
\]

Thus, we see \( f_0 \in \mathcal{F}^+ \) and \( T_0 \in \mathcal{T}^+ \).

Now, from Lemma 13,

\[
\frac{dQ}{dP} = \arg \min_{f(x) \in \mathcal{F}^+} \int \sqrt{f(x)} \frac{dP}{d\mu} d\mu + \int \frac{1}{\sqrt{f(x)}} \frac{dQ}{d\mu} d\mu
\]

holds, and it follows from (60) that

\[
\min_{f \in \mathcal{F}^+} \int \sqrt{f(x)} \frac{dP}{d\mu} d\mu + \int \frac{1}{\sqrt{f(x)}} \frac{dQ}{d\mu} d\mu = \min_{T \in \mathcal{T}^+} \int \sqrt{Z_T} e^{-T(X)} \frac{dP}{d\mu} d\mu + \int \frac{1}{Z_T e^{T(X)}} \frac{dQ}{d\mu} d\mu
\]

\[
= \min_{T \in \mathcal{T}^+} \left\{ \sqrt{\frac{1}{Z_T}} e^{-T(X)} \frac{dP}{d\mu} + \frac{1}{Z_T} e^{T(X)} \frac{dQ}{d\mu} \right\} d\mu,
\tag{60}
\]

where \( Z_T = E_P[e^{-T(X)}] \) and \( Z_T' = E_Q[e^{T(X)}] \).

Now, we note that

\[
\sqrt{f_0} \frac{dP}{d\mu} = \sqrt{\frac{dQ}{dP}} d\mu
\]

\[
= \sqrt{\frac{dP}{dQ}} d\mu
\]

\[
= \sqrt{f_0}^{-1} \frac{dQ}{d\mu}
\]

24
holds. In addition, since $\alpha + \beta \geq 2\sqrt{\alpha\beta}$ holds for any $\alpha > 0$ and $\beta > 0$ and the equality holds when $\alpha = \beta$, we also note that it holds that
\[
\sqrt{f_0} \frac{dP}{d\mu} + \sqrt{(f_0)^{-1}} \frac{dQ}{d\mu} = 2 \sqrt{\sqrt{f_0} \frac{dP}{d\mu} \cdot \sqrt{(f_0)^{-1}} \frac{dQ}{d\mu}}. 
\] (61)

Now, let $T^*$ be the minimizer for (60), then we see
\[
f_0(X) = \frac{1}{Z_{T^*}} e^{-T^*(X)}, \quad \text{where} \quad Z_{T^*} = E_P[e^{-T^*(X)}]. \tag{62}
\]

Inserting (62) to (61), it holds that
\[
\sqrt{\frac{1}{Z_{T^*}} e^{-T^*(X)}} \frac{dP}{d\mu} + \sqrt{\frac{1}{Z_{T^*}} e^{T^*(X)}} \frac{dQ}{d\mu} = 2 \sqrt{\frac{1}{Z_{T^*}} e^{-T^*(X)}} \frac{dP}{d\mu} \cdot \sqrt{\frac{1}{Z_{T^*}} e^{T^*(X)}} \frac{dQ}{d\mu},
\]
where $Z_{T^*} = E_P[e^{-T^*(X)}]$ and $Z'_{T^*} = E_Q[e^{T^*(X)}]$. Thus,
\[
\min_{T \in T^*} \int \left\{ \frac{1}{Z_T} e^{-T(X)} \frac{dP}{d\mu} + \frac{1}{Z_T} e^{T(X)} \frac{dQ}{d\mu} \right\} d\mu
\]
\[
= \min_{T \in T^*} \int 2 \sqrt{\frac{1}{Z_T} e^{-T(X)} \frac{dP}{d\mu} \cdot \frac{1}{Z_T} e^{T(X)} \frac{dQ}{d\mu}} d\mu
\]
\[
= \min_{T \in T^*} \int \frac{2}{(Z_T Z'_{T})^{\frac{1}{2}}} \sqrt{\frac{dP}{d\mu} \cdot \frac{dQ}{d\mu}} d\mu
\]
\[
= \left( 2 \int \frac{dP}{d\mu} \cdot \frac{dQ}{d\mu} \right) \min_{T \in T^*} \left\{ \frac{1}{(Z_T Z'_{T})^{\frac{1}{2}}} \right\}^{\frac{1}{2}}
\]
\[
= \left( 2 \int \frac{dP}{d\mu} \cdot \frac{dQ}{d\mu} \right) \left\{ \max_{T \in T^*} Z_T Z'_{T} \right\}^{-\frac{1}{2}} \tag{63}
\]

From (60) and (63), we see
\[
T_0 = \arg\min_{T \in T^*} \int \left\{ \frac{1}{Z_T} e^{-T(X)} \frac{dP}{d\mu} + \frac{1}{Z_T} e^{T(X)} \frac{dQ}{d\mu} \right\} d\mu,
\]
\[
= \arg\max_{T \in T^*} Z_T Z'_{T}
\]
\[
= \arg\max_{T \in T^*} E_P[e^{-T(X)}] \cdot E_Q[e^{T(X)}]. \tag{64}
\]
Since \( \frac{dQ}{dP} = f_0 = \frac{1}{Z_{T_0}} e^{-T_0(X)} \) and \( Z_{T_0} = E_P[e^{-T_0(X)}] \), we have

\[
\frac{dQ}{dP} = \frac{1}{Z_{T_0}} e^{-T_0}
\]

and

\[
\frac{dP}{dQ} = \frac{1}{Z_{T_0}'} e^{T_0},
\]

where \( Z_{T_0} = E_P[e^{-T_0}] \) and \( Z_{T_0}' = E_Q[e^{T_0}] \). Finally, we show \( T \) which minimizes the right side term of (57) is only \( T_0 \). Let

\[
T_1 = \arg \max_{T \in T} E_P[e^{-T(X)}] \cdot E_Q[e^{T(X)}].
\]

From the above discussion, we have

\[
\frac{dQ}{dP} = \frac{1}{Z_{T_1}} e^{-T_1}
\]

and

\[
\frac{dP}{dQ} = \frac{1}{Z_{T_1}'} e^{T_1},
\]

where \( Z_{T_1} = E_P[e^{-T_1}] \) and \( Z_{T_1}' = E_Q[e^{T_1}] \). It follows from (65) and (68) that

\[
E_P \left[ \phi(X) \left( \frac{1}{Z_{T_1}} e^{-T_1(X)} - \frac{1}{Z_{T_0}} e^{-T_0(X)} \right) \right] = 0,
\]

for any positive measurable function \( \phi(X) \). Thus, we see

\[
\frac{1}{Z_{T_1}} e^{-T_1(X)} = \frac{1}{Z_{T_0}} e^{-T_0(X)} \quad \text{a.e. } x \text{ for } P,
\]

then it holds that \( T_0(x) = T_1(x) \) a.e \( x \) for \( P \). Similarly, it follows from (65) and (69) that \( T_0(x) = T_1(x) \) a.e \( x \) for \( Q \). Then, we have

\[
T_0(x) = T_1(x) \quad \text{a.e. } x \text{ for } \mu.
\]

This completes the proof. \( \square \)

**Lemma 15.** Let \( G \) be a DAG for \( V \) and \( U \). Suppose that \( P(Y|do(X), Z) \) is identifiable in \( G \). For disjoint sets \( X, Y, Z \subset V \), let \( X \subset An(Y)_G \) and \( Z_{De} = Z \cap De(X)_G \cap De(Y)_G \). Then,

\[
P(Y|do(X), Z) = \begin{cases} P(Y|X, Z) & \text{if } Z_{De} = \phi, \\ \frac{P(Y|X, Z, D_{De})P(D_{De}|Y, X, Z, D_{De})}{P(D_{De}|X, Z, D_{De})} & \text{if } Z_{De} \neq \phi. \end{cases}
\]

26
proof of Lemma 15. In this proof, each \( Z_i \in \mathbf{Z} \) is assumed not independent of \( \mathbf{Y} \). If there exists \( Z_i \in \mathbf{Z} \) such that it is independent of \( \mathbf{Y} \), remove them by applying do-calculus R1. Then, we can prove this case in the same way below.

Firstly, we note that \( \mathbf{Z} \cap \text{An}(\mathbf{X})_G \cap \text{De}(\mathbf{Y})_G = \phi \). Let \( \mathbf{V}' = \mathbf{V} \setminus (\mathbf{Y} \cup \mathbf{X} \cup \mathbf{Z}) \). If \( \mathbf{Z} \cap \text{An}(\mathbf{X})_G \cap \text{De}(\mathbf{Y})_G \neq \phi \), then there exists a directed path such that \( \mathbf{Y} \rightarrow^{\mathbf{V}'} \mathbf{Z} \rightarrow^{\mathbf{V}'} \mathbf{X} \). Here, \( \rightarrow^{\mathbf{V}'} \) denote a path through only variables of \( \mathbf{V}' \).

This contradicts the assumption \( \mathbf{X} \subset \text{An}(\mathbf{Y})_G \). Thus, \( \mathbf{Z} \) can be divided into three parts: \( \mathbf{Z} = \mathbf{Z}_1 \cup \mathbf{Z}_2 \cup \mathbf{Z}_3 \), where \( \mathbf{Z}_1 \), \( \mathbf{Z}_2 \) and \( \mathbf{Z}_3 \) are disjoint sets such that

\[
\begin{align*}
\mathbf{Z}_1 &= (\mathbf{Z} \setminus \text{De}(\mathbf{X})_G) \cap \text{An}(\mathbf{Y})_G, \\
\mathbf{Z}_2 &= \mathbf{Z} \cap \text{De}(\mathbf{X})_G \cap \text{An}(\mathbf{Y})_G, \\
\mathbf{Z}_3 &= \mathbf{Z} \cap \text{De}(\mathbf{X})_G \cap \text{De}(\mathbf{Y})_G.
\end{align*}
\]

Secondly, we note that if there exist the directed paths through \( \mathbf{V}' \) between \( \mathbf{Z}_1 \), \( \mathbf{Z}_2 \) and \( \mathbf{Z}_3 \), then each of the paths falls into one of the following three cases:

1. \( \mathbf{Z}_1 \rightarrow^{\mathbf{V}'} \mathbf{Z}_2 \)
2. \( \mathbf{Z}_2 \rightarrow^{\mathbf{V}'} \mathbf{Z}_3 \)
3. \( \mathbf{Z}_1 \rightarrow^{\mathbf{V}'} \mathbf{Z}_3 \)

In fact, as to P1 for example, if there exists a directed path through \( \mathbf{Z}_2 \rightarrow^{\mathbf{V}'} \mathbf{Z}_1 \), then there also exists a path such that \( \mathbf{X}_i \rightarrow^{\mathbf{V}'} \mathbf{Z}_2 \rightarrow^{\mathbf{V}'} \mathbf{Z}_1 \). This contradicts the assumption \( \mathbf{Z}_1 \subset \mathbf{Z} \setminus \text{De}(\mathbf{X})_G \). Similarly, the other paths than listed above are also denied.

Therefore, we can obtain \( P(\mathbf{Y}, \mathbf{X}, \mathbf{Z}) = \sum_{\mathbf{X}_V} P(\mathbf{V}) \) as

\[
P(\mathbf{Y}, \mathbf{X}, \mathbf{Z}) = P(\mathbf{Y}|\mathbf{X}, \mathbf{Z}_1, \mathbf{Z}_2) \cdot P(\mathbf{X}|\mathbf{Z}_1) \cdot P(\mathbf{Z}_1) \times P(\mathbf{Z}_2|\mathbf{X}, \mathbf{Z}_1) \cdot P(\mathbf{Z}_3|\mathbf{Y}, \mathbf{X}, \mathbf{Z}_1, \mathbf{Z}_2).
\]

By (1), we have

\[
P(\mathbf{Y}, \mathbf{Z}|do(\mathbf{X})) = P(\mathbf{Y}|\mathbf{X}, \mathbf{Z}_1, \mathbf{Z}_2) \cdot P(\mathbf{Z}_1) \cdot P(\mathbf{Z}_2|\mathbf{X}, \mathbf{Z}_1) \times P(\mathbf{Z}_3|\mathbf{Y}, \mathbf{X}, \mathbf{Z}_1, \mathbf{Z}_2).
\]  \hspace{1cm} (73)

In the case \( \mathbf{Z}_3 = \phi \), by marginalizing out \( \mathbf{Y} \) of (73), we have

\[
P(\mathbf{Z}|do(\mathbf{X})) = \sum_{\mathbf{Y} \in \mathbf{X}_V} P(\mathbf{Y} = \mathbf{y}, \mathbf{Z}|do(\mathbf{X}))
\]

\[
= \sum_{\mathbf{Y} \in \mathbf{X}_V} P(\mathbf{Y} = \mathbf{y}|\mathbf{X}, \mathbf{Z}_1, \mathbf{Z}_2) \cdot P(\mathbf{Z}_1) \cdot P(\mathbf{Z}_2|\mathbf{X}, \mathbf{Z}_1)
\]

\[
= P(\mathbf{Z}_1) \cdot P(\mathbf{Z}_2|\mathbf{X}, \mathbf{Z}_1) \sum_{\mathbf{Y} \in \mathbf{X}_V} P(\mathbf{Y} = \mathbf{y}|\mathbf{X}, \mathbf{Z}_1, \mathbf{Z}_2)
\]

\[
= P(\mathbf{Z}_1) \cdot P(\mathbf{Z}_2|\mathbf{X}, \mathbf{Z}_1).
\]
On the other hand, in the case that $Z_3 \neq \phi$, we obtain

$$P(Z|do(X)) = \sum_{y \in \mathcal{Y}} P(Y = y, Z|do(X))$$

$$= \sum_{y \in \mathcal{Y}} P(Y = y|X, Z_1, Z_2) \cdot P(Z_1)$$

$$\times P(Z_2|X, Z_1) \cdot P(Z_3|Y = y, X, Z_1, Z_2)$$

$$= P(Z_1) \cdot P(Z_2|X, Z_1) \sum_{y \in \mathcal{Y}} P(Z_3|Y = y, X, Z_1, Z_2)$$

$$\times P(Y = y|X, Z_1, Z_2)$$

$$= P(Z_1) \cdot P(Z_2|X, Z_1) \cdot P(Z_3|X, Z_1, Z_2).$$

Summarizing the above results, we have

$$P(Z|do(X)) = \begin{cases} P(Z_1) \cdot P(Z_2|X, Z_1), & \text{if } Z_3 = \phi, \\ P(Z_1) \cdot P(Z_2|X, Z_1) \cdot P(Z_3|X, Z_1, Z_2) & \text{if } Z_3 \neq \phi. \end{cases} \tag{74}$$

Inserting (73) and (74) into (2), we finally obtain

$$P(Y|do(X), Z) = \frac{P(Y, Z|do(X))}{P(Z|do(X))}$$

$$= \begin{cases} \frac{P(Y|X, Z_1, Z_2)}{P(Z_1) \cdot P(Z_2|X, Z_1)} & \text{if } Z_3 = \phi, \\ \frac{P(Y|X, Z_1, Z_2)}{P(Z_1) \cdot P(Z_2|X, Z_1)} & \text{if } Z_3 \neq \phi. \end{cases} \tag{75}$$

Note that $Z_{De} = Z_3$ and $Z \setminus Z_{De} = Z_1 \cup Z_2$, therefore we obtain (72).

This completes the proof. \hfill \Box

**Theorem 16 (Theorem 8 restated).** Given disjoint sets of $X = \{X_1, X_2, \ldots, X_n\}$, $Y$, $Z \subseteq V$ which satisfy

$$X = \{X_1, X_2, \ldots, X_n\} \subseteq An(Y)_G \quad \text{and} \quad Z \cap De(X)_G \cap De(Y)_G = \phi. \tag{75}$$

Let $P = P(X_1, X_2, \ldots, X_n)$ and $Q = P(X_1) \times P(X_2) \times \cdots \times P(X_n) \times P(Z)$. Suppose that $P$ and $Q$ are equivalent, and satisfy

$$E_P \left[ \left( \frac{dQ}{dP} \right)^{\frac{3}{2}} \right] < \infty \quad \text{and} \quad E_Q \left[ \left( \frac{dP}{dQ} \right)^{\frac{3}{2}} \right] < \infty. \tag{76}$$

Let $\tilde{P} = P(Y|do(X), Z) \times P(X_1) \times P(X_2) \times \cdots \times P(X_n) \times P(Z)$. Then, let

$$T^*(X_1, X_2, \ldots, X_n, Z) = \arg \sup_{T(x, z): \Omega \rightarrow \mathbb{R}} E_P e^{-T} \cdot E_Q [e^{T}], \tag{77}$$

where the supremum is taken over all functions $T$ such that both $E_P e^{T(X, Z)/2}$ and $E_Q e^{-T(X, Z)/2}$ are finite. For $T^*$, it holds that

$$\frac{d\tilde{P}}{dP} = \frac{1}{Z} e^{-T^*(X_1, X_2, \ldots, X_n, Z)} \tag{78}$$
and
\[
\frac{dP}{dP} = \frac{1}{Z} e^{T^*(X_1, X_2, \ldots, X_n, Z)},
\]  
(79)

where \( Z = E_P [e^{-T^*(X_1, X_2, \ldots, X_n, Z)}] \) and \( \tilde{Z} = E_Q [e^{T^*(X_1, X_2, \ldots, X_n, Z)}] \).

**proof of Theorem 16.** From Lemma 15 and the assumption (75),
\[
\tilde{Z} = P(Y | \text{do}(X), Z) \times P(X_1) \times P(X_2) \times \cdots \times P(X_n) \times P(Z)
\]
Thus, from Theorem 14, it holds that
\[
\frac{1}{Z} e^{-T^*(X_1, X_2, \ldots, X_n, Z)} = \frac{dQ}{dP}
\]
(80)

and
\[
\frac{1}{Z} e^{T^*(X_1, X_2, \ldots, X_n, Z)} = \frac{dP}{dQ}.
\]
(81)

where \( Z = E_P [e^{-T^*(X_1, X_2, \ldots, X_n, Z)}] \) and \( \tilde{Z} = E_Q [e^{T^*(X_1, X_2, \ldots, X_n, Z)}] \).

This completes the proof.

**Theorem 17 (Theorem 10 restated).** For a pair of equivalent probability measures \( P \) and \( Q \) which satisfy
\[
E_P \left[ \left( \frac{dQ}{dP} \right)^\frac{3}{2} \right] < \infty \quad \text{and} \quad E_Q \left[ \left( \frac{dP}{dQ} \right)^\frac{3}{2} \right] < \infty,
\]
the Kullback-Leibler divergence between \( P \) and \( Q \) is given as
\[
D_{KL}(P\|Q) = - \sup_{T(x) : \Omega \to \mathbb{R}} \log \left( E_P[e^{-T(X)}] \cdot E_Q[e^{T(X)}] \right),
\]
(83)

where the supremum is taken over all functions \( T \) such that \( E_P[e^{-T(X)/2}] \) and \( E_Q[e^{-T(X)/2}] \) are finite. In addition, the above equality holds for \( T^* \) such that
\[
\frac{dQ}{dP} = \frac{1}{Z} e^{-T^*}
\]
(84)

and
\[
\frac{dP}{dQ} = \frac{1}{Z'} e^{T^*},
\]
(85)

where \( Z = E_P [e^{-T^*(X)}] \) and \( Z' = E_Q [e^{T^*(X)}] \).

**proof of Theorem 17.** First, from the definition of Kullback-Leibler divergence, we have
\[
D_{KL}(P\|Q) = \int dP \log \left( \frac{dP}{dQ} \right)
\]
\[
= \int (dP + dQ) \log \left( \frac{dP}{dQ} \right) - \int dQ \log \left( \frac{dP}{dQ} \right)
\]
\[
= \int (dP + dQ) \log \left( \frac{dP}{dQ} \right) + \int dQ \log \left( \frac{dQ}{dP} \right). \]  
(86)
By theorem 7, replacing $\frac{dQ}{dP}$ and $\frac{dP}{dQ}$ with

$$\frac{dQ}{dP} = \frac{1}{Z} e^{-T^*(x)} \quad \text{and} \quad Z = E_P[e^{-T^*(x)}],$$

and

$$\frac{dP}{dQ} = \frac{1}{Z'} e^{T^*(x)} \quad \text{and} \quad Z' = E_Q[e^{T^*(x)}],$$

then we obtain

$$\int (dP + dQ) \log \left( \frac{dP}{dQ} \right) + \int dQ \log \left( \frac{1}{Z} e^{T^*(x)} \right) = E_P [-T^*(x)] + E_Q [-T^*(x)] - 2 \log(Z) + E_Q [T^*(X)] - \log(Z')$$

$$= -E_P [T^*(X)] - 2 \log(Z) - \log(Z').$$  \hfill (87)

Note that, by Theorem 10, it holds that

$$E_P [T^*(X)] = D_{KL}(P\|Q) + \log \left( E_Q \left[ e^{T^*(X)} \right] \right) = D_{KL}(P\|Q) + \log Z'.$$  \hfill (88)

Finally, from (86), (87) and (88), we have

$$D_{KL}(P\|Q) = \int (dP + dQ) \log \left( \frac{dP}{dQ} \right) + \int dQ \log \left( \frac{dQ}{dP} \right)$$

$$= -E_P [T(X)] - 2 \log(Z) - \log(Z')$$

$$= -D_{KL}(P\|Q) - \log(Z') - 2 \log(Z) - \log(Z').$$

Therefore,

$$D_{KL}(P\|Q) = -\log(Z) - \log(Z') = -\log \left( E_P[e^{-T^*(X)}] \cdot E_Q[e^{T^*(X)}] \right).$$

This completes the proof.

A.2 Back-Propagation Algorithm via NGD weights

We show a back-propagation algorithm with our stable weights for the MSE loss in Algorithm 3. The MSE loss for the back-propagation is calculated from the mean of the element wise product of squared errors and balancing weights.

A.3 Neural Network and Training Details in Experiment 1 and 2

In Table 3 we show a neural network we used for building of NGD models in Experiment 1 and Experiment 2. Both of the neural networks for the two experiments are the same, which have 8 hidden layers with 20 units of each layer. For training them, we used the Adam algorithm of Pytorch. The hyperparameters we used for training NGD models are provided in Table 4.
Algorithm 3 Back-Propagation for Mean Squared Error Loss

**Input:** Data: \((y, x_1, x_2, \ldots, x_n, z) = \{(y^i, x_{1}^i, x_{2}^i, \ldots, x_{n}^i, z^i) | i = 1, 2, \ldots, N\},

A NGD model: \(T\)

**Output:** A trained neural network model \(f_\phi\) for \(E_{\tilde{P}}[Y|X, Z]\)

1: repeat
2: \(\hat{y} \leftarrow f_\phi(x_1, x_2, \ldots, x_n, z)\) \(\triangleright\) Forward Propagation
3: \(SW(x_1, x_2, \ldots, x_n, z) \leftarrow \frac{e^{-T(x_1, x_2, \ldots, x_n, z)}}{\text{MEAN}(e^{-T(x_1, x_2, \ldots, x_n, z)})}\)
4: \(\triangleright\) Calculate stable weights
5: \(\text{Err}_\phi \leftarrow y - \hat{y}\) \(\triangleright\) Calculate Error \(\text{Err}_\phi\)
6: \(L_\phi \leftarrow \text{MEAN}(\text{Err}_\phi \otimes \text{Err}_\phi \otimes SW(x_1, x_2, \ldots, x_n, z))\)
7: \(\triangleright\) Calculate Loss \(L_\phi\)
8: \(\phi \leftarrow \phi - \nabla L_\phi\) \(\triangleright\) Update the parameters of \(f_\phi\)
9: until convergence

Table 3: Network for building of NGD models in Experiment 1 and 2

| Layer                        | Number of outputs | Activation function |
|------------------------------|-------------------|---------------------|
| Input x                      | 20                |                     |
| Dense(hidden layer 1 ~ 8)    | 20                | ReLU                |
| Output                       | 1                 | Linear              |

Table 4: Hyperparameters for training NGD models in Experiment 1 and 2

| Learning rate | 0.001 |
|---------------|-------|
| MiniBatch size| 2000  |
| Number of epochs| 600   |