Towards All-Purpose Domain Adaptation Under Confounding

Calvin McCarter
mccarter.calvin@gmail.com

Abstract

Current domain adaptation methods address the problems of covariate shift or label shift, but are not applicable to the setting where they occur simultaneously and interact with each other. In this paper, we propose a new assumption, confounded shift, to begin to address this problem. We also propose a framework for this task, based on minimizing the expected divergence between the source and target conditional distributions. Within this framework, we propose using the reverse KL divergence, demonstrating the use of both parametric linear Gaussian and nonparametric nonlinear Gaussian Process estimators of the conditional distribution. We also propose using the Maximum Mean Discrepancy (MMD) within our framework. To make confounded domain adaptation with the MMD effective, we propose an intelligent dynamic strategy for choosing the kernel bandwidth, which may be of independent interest even outside of the confounded shift context. Finally, we show that our approach is advantageous on a variety of synthetic and real datasets.

1 Introduction

Suppose you have developed a seizure risk prediction model using electroencephalogram (EEG) data, but your hospital lab recently acquired an updated EEG machine. Based on the small amount of data collected for validating the V2 machine, it appears that the V2 machine data distribution is shifted relative to that from the V1 machine. At this point, the problem might appear to call for the use of covariate shift domain adaptation approaches, to adapt the V2 (source) distribution to look like the V1 (target) distribution. Yet additionally, while the V1 dataset comes from a large number of low-risk and high-risk individuals, the V2 dataset thus far is mostly comprised of low-risk volunteers. Ignoring the aforementioned covariate shift problem, this latter problem may seem to fall into the label shift domain adaptation problem setting. However, our hypothetical scenario combines these two problems: it has both covariate shift and label shift which are confounded with each other.

In the above scenario, the label variable (seizure risk) was coincidentally also the confounder that was correlated with the V1-vs-V2 batch effect. But we might also want to perform statistical analyses for scientific purposes on the EEG data, after combining data from both V1 and V2 machines, to increase our statistical power. For example, we might want to correct for the risk-machine confounding, and then use the adapted EEG data to predict age, in order to discover EEG features related to aging processes. Or, using a corrected and combined dataset, we might want to predict EEG data given medication status, to see how certain medications affect EEG features.

Therefore, in this paper we seek a domain adaptation method that creates a “general-purpose” fix for the source-vs-target batch effect in our data. In other words, we would like to either adapt the V1 data to look like V2 data, or vice versa. Then we would combine these two datasets, and use them as one dataset for a variety of downstream prediction and inference tasks.

There are a few obstacles to solving this problem. First, even for tasks that do not involve predicting the confounder (e.g. seizure risk), we cannot simply perform standard batch correction approaches,
because the source and target datasets should not look alike. Second, even for these other tasks that do not involve predicting the confounder (e.g. seizure risk), we cannot assume that the confounder (seizure risk) is known for all samples on which we will apply our adaptation. Therefore, we need an adaptation function (e.g. which corrects for the V1-versus-V2 shift) that does not take in the confounder as an input feature (e.g. which does not depend on seizure risk). Third, we cannot discard all information unrelated to predicting the confounder. One common approach for domain adaptation is to learn an intermediate representation that is invariant to source-vs-target effects, while still predictive of the label (which is the confounder). But because we want to use adapted-and-combined dataset for a variety of downstream tasks, we need to preserve as much information as possible, not merely the subspace relevant to label prediction.

To address this challenge we describe a unique problem setting in domain adaptation: **confounded shift**. Confounded shift does not assume that the confounding variable(s) are identically distributed in the source and target domains, or that the covariates are identically distributed in the source and target domains. Rather, it assumes that there exists an adaptation $g(X)$ from source covariates to target covariates such that the target’s conditional distribution of covariates given confounders is equal to that of the adapted-source’s conditional distribution. However, we do not assume that the adapted-source’s covariates and target’s covariates have the same distribution.

In the rest of the paper, we provide a framework for adapting the source to the target, based on minimizing the expected divergence between target and adapted-source conditional distributions, i.e. conditioning on the confounding variables. We examine three different divergence functions, the forward-KL divergence, the reverse-KL divergence, and the maximum mean discrepancy (MMD).

Furthermore, using this framework we provide concrete implementations based on the assumption that the source-vs-target batch effect is “simple” while the relationship between covariates and confounder is potentially complex. Therefore, we restrict the adaptation to be linear, and potentially even a diagonal matrix. Meanwhile, we consider both simple (multivariate linear Gaussian) and complex (Gaussian Process and sampling-based) estimators for the conditional distribution of the covariates given the confounder(s). We make this assumption because our method is intended to adapt structured input data, such as biometric sensor outputs, genomic sequencing data, and financial market data, where batch effects are typically simple, while the input-output mapping is often nonlinear. We are not, for example, attempting to adapt an image classification model from photographic inputs to hand-drawn inputs, which is a motivating example for much recent work on domain adaptation.

## 2 Preliminaries

In this section, we introduce our notation, provide an overview of previous related domain adaptation settings, and describe (standard) linear domain adaptation.

### 2.1 Notation

Our notation is inspired by the setting where the confounding variable is the label variable, even though our framework is not strictly intended for this scenario. $X$ and $Y$ respectively denote the covariate (input feature) and confounder (output label) space. $X$ and $Y$ denote random variables which take values in $X$ and $Y$, respectively. A joint distribution over covariate space $X$ and confounder space $Y$ is called a domain $D$. In our setting, there is a source domain $D_S$ and a target domain $D_T$. $D_X^S$, $D_Y^S$ denote the marginal distributions of covariates under the source and target domains, respectively; $D_X^T$, $D_Y^T$ denote the corresponding marginal distributions of confounders. For arbitrary distributions $P$ and $Q$, we assume we have been given a distance or divergence function denoted by $d(P, Q)$. By $N(\mu, \Sigma)$ we denote the Gaussian distribution with mean $\mu$ and covariance $\Sigma$.

### 2.2 Background on Linear Domain Adaptation

Domain adaptation has a closed form solution in the special case of two multivariate Gaussian distributions. The optimal transport (OT) map under the type-2 Wasserstein metric for $x \sim N(\mu_Q, \Sigma_Q)$ to a different Gaussian distribution $N(\mu_P, \Sigma_P)$ has been shown (Dowson & Landau 1982; Knott & Smith 1984) to be the following:

$$x \mapsto \mu_P + A(x - \mu_Q) = Ax + (\mu_P - A\mu_Q), \quad (1)$$
Table 1: Domain adaptation settings

| Name                    | Shift                        | Assumed Invariant |
|-------------------------|------------------------------|-------------------|
| Covariate Shift         | \(D^X_S \neq D_T^X\)         |                   |
| Label Shift             | \(D^Y_S \neq D_T^Y\)         |                   |
| Generalized Label Shift | \(D^S_X \neq D^T_X\)         |                   |
| Confounded Shift        | \(D^S_Y \neq D_T^Y\)         |                   |

where

\[
A = \Sigma_Q^{-1/2} \left( \Sigma_P^{1/2} \Sigma_P^{1/2} \right)^{1/2} \Sigma_Q^{-1/2} = A^\top.
\]

(2)

For univariate Gaussians \(N(\mu_Q, \sigma_Q^2)\) and \(N(\mu_P, \sigma_P^2)\), the above transformation simplifies to

\[
x \mapsto \mu_P + \frac{\sigma_P}{\sigma_Q} (x - \mu_Q) = \frac{\sigma_P}{\sigma_Q} x + (\mu_P - \frac{\sigma_P}{\sigma_Q} \mu_Q).
\]

(3)

This mapping has been applied to a variety of uses [Mallasto & Feragen, 2017; Muzellec & Cuturi, 2018; Shafieezadeh Abadeh et al., 2018; Peyré et al., 2019] in optimal transport and machine learning.

2.3 Background on Covariate Shift, Label Shift, and Generalized Label Shift

Domain adaptation methods typically assume either covariate shift or label shift. With covariate shift, the marginal distribution over covariates differs between source and target domains. However, for any particular covariate, the conditional distribution of the label given the covariate is identical between source and target. With label shift, the marginal distribution over labels differs between source and target domains. However, for any particular label, the conditional distribution of the covariates given the label is identical between source and target domain.

More recently, generalized label shift was introduced to allow covariate distributions to differ between source and target domains [Tachet des Combes et al., 2020]. Generalized label shift (GLS) instead assumes that, given a transformation function \(Z = g(X)\) applied to inputs from both source and target domains, the conditional distributions of \(Z\) given \(Y = y\) are identical for all \(y\). This is a weak assumption, and it applies to our problem setting as well. However, it is designed for the scenario where we simply need \(g\) to preserve information only for predicting \(Y\) given \(X \sim D^T_X\).

3 Our Assumption: Confounded Shift

In our case, given \(X \sim D^X_S\), we instead want to recover what it would have been had we observed the same object from the data generating process corresponding to the target domain \(X \sim D^X_T\). In other words, the mapping \(g(X)\) should not only preserve information in \(X\) useful for predicting \(Y\), but ideally all information in \(X \sim D^X_S\) that is contained in \(X \sim D^X_T\).

Relation to Generalized Label Shift Suppose GLS intermediate representation \(g(X)\) were extended to be a function of both \(X\) and an indicator variable \(D\) specifying whether a sample is taken from the target or the source domain. Then, given this extended representation \{\(X, D\}\), we restrict \(\tilde{g}(\{X, D\})\) as follows,

\[
\tilde{g}(\{X, D\}) = \begin{cases} g(X) & D = S \\ X & D = T \end{cases}
\]

(4)

so that samples from the source distribution are adapted by \(g(\cdot)\), while those from the target distribution pass through unchanged. With this extended representation, as well as the restriction on \(\tilde{g}\), confounded shift and GLS coincide. Note that while confounded shift is stronger than GLS, both allow \(D^S_S \neq D^X_S\); and just as GLS allows \(D^g(X) \neq D^P(X)\), we analogously allow \(D^g(X) \neq D^T_T\). The previous assumptions as well as our confounded shift assumption are summarized in Table 1.
4 Our Approach: Confounded Domain Adaptation

For the time being, we will consider our motivating scenario in which our ultimate goal is to reuse a minimum-risk hypothesis \( h : \mathcal{X} \rightarrow \mathcal{Y} \) in a new deployment setting. We treat the deployment setting as the source domain, instead of (as is typical) the target domain. And instead of learning an end-to-end predictor for the deployment domain, we learn an adaptation from it to the target domain for which we have a large number of labelled examples. Then, to perform predictions on the deployment (source) domain, we first adapt them to the target domain, and then we apply prediction model trained on the target domain. In other words, we do not need to retrain \( h \), and instead apply \( h \circ g \) to incoming unlabelled source samples. Similarly, other prediction tasks and statistical analyses can be identically applied to target domain data and adapted-source domain data.

We assume the existence of a noise-free mapping from the deployment domain to the large labelled dataset domain. In real-world structured data applications, new data sources are designed with “backwards-compatibility” in mind, in the hope that existing models may be reused. However, our framework could instead be applied while treating the deployment setting as the target domain and the large labelled dataset as the source domain. This entails retraining \( h \) on adapted data; we then apply \( h \) to new samples.

We assume \( N_S \) and \( N_T \) samples from the source and target domain, respectively, with \( N_S < N_T \). We assume each sample feature vector \( \mathbf{x}^{(n)} \in \mathbb{R}^M \). To accommodate both categorical and continuous confounders (or a concatenation of both), they will generally be accessed via kernel function \( k_Y(y^{(n_1)}, y^{(n_2)}) \).

### 4.1 Main Idea

We propose to minimize the expected distance (or divergence) \( d \) between the conditional distributions of source and target given confounders, under some specified prior distribution over the confounders. Our goal is to find the optimal linear transformation \( g(x) = Ax + b \) of the source to target, leading to the following objective:

\[
\min_{A, b} E_{y \sim D^Y_T} d \left( D_T(x|Y = y), D_S(Ax + b|Y = y) \right). 
\]

(5)

In certain scenarios, particularly scientific analyses, it is important for explainability that each \( r \)th adapted feature \( \{Ax^{(n)} + b\} \), be derived only from the original feature \( \{x^{(n)}\} \). So we will also examine the case where \( A \) is restricted to be diagonal \( A = \text{diag}(a) \); this is sometimes referred to as a location-scale adaptation (Zhang et al., 2013).

Three ingredients remain to turn this framework into a concrete algorithm: the distance/divergence function \( d \) (Section 4.2), the conditional distribution estimators \( D(\cdot|Y = y) \) (Section 4.3), and the choice of prior confounder distribution over which to minimize risk \( D^X_T \) (Section 4.4).

### 4.2 Distribution Distance/Divergence Function

#### 4.2.1 Forward and Reverse KL-divergences

It can be straightforwardly shown that the linear map Eq. (1) derived from optimal transport leads to adapted data being distributed according to the target distribution. That is, \( \mu_P + A(x - \mu_Q) \sim N(\mu_P, \Sigma_P) \). Therefore, the KL-divergence from the target distribution to the adapted source data distribution is minimized to 0, and similarly for the KL-divergence from the adapted source data distribution to the target distribution. This motivates using the KL-divergence as a divergence measure to minimize. Minimizing the forward KL-divergence from target to adapted-source is the natural choice in this setting. However, we would expect the reverse KL-divergence to be more computationally tractable and well-conditioned, motivating its use in variational inference (Blei et al., 2012) and reinforcement learning (Kappen et al., 2012; Levine, 2018). Thus, we choose between \( d(P, Q) := d_{KL}(P||Q) \) (forward-KL) or \( d(P, Q) := d_{KL}(Q||P) \) (reverse-KL). In either case, it can be shown that minimizing Eq. (5) requires estimating the conditional means and conditional covariances, according to both the source and target domain estimators, evaluated at each \( y \sim D^Y_T \).
4.2.2 Maximum Mean Discrepancy

An alternative approach is to represent the distance between target and adapted-source conditional distributions as the distance between (conditional) mean embeddings. This gives rise to using the maximum mean discrepancy (MMD) as our distance function, where our MMD is defined by a feature map \( \phi \) mapping features \( x \in \mathcal{X} \) to a reproducing kernel Hilbert space \( \mathcal{H} \). We denote our feature-space kernel based on \( \phi \) as \( k_{\mathcal{X}}(x^{(n)}_1, x^{(n)}_2) = \langle \phi(x^{(n)}_1), \phi(x^{(n)}_2) \rangle \). For a particular \( y \in \mathcal{Y} \), we assume we have a way of sampling from \( \mathcal{D}_T(\cdot | Y = y) \) and \( \mathcal{D}_S(\cdot | Y = y) \). Then, we have

\[
d\left( \mathcal{D}_T(\cdot | Y = y), \mathcal{D}_S(\cdot | Y = y) \right) := \text{MMD}^2(\mathcal{D}_T(\cdot | Y = y), \mathcal{D}_S(\cdot | Y = y))
\]

\[
= \mathbb{E}_{x^{(n)}_1, x^{(n)}_1' \sim \mathcal{D}_T(\cdot | Y = y)} k_{\mathcal{X}}(x^{(n)}_1, x^{(n)}_1')
- 2\mathbb{E}_{x^{(n)}_1 \sim \mathcal{D}_T(\cdot | Y = y), x^{(n)}_2 \sim \mathcal{D}_S(\cdot | Y = y)} k_{\mathcal{X}}(x^{(n)}_1, Ax^{(n)}_2 + b)
+ \mathbb{E}_{x^{(n)}_2, x^{(n)}_2' \sim \mathcal{D}_S(\cdot | Y = y)} k_{\mathcal{X}}(Ax^{(n)}_2 + b, Ax^{(n)}_2' + b).
\]

We efficiently minimize this objective by sampling batches from the conditional distributions, combined with (batch) gradient descent with momentum.

The MMD distance function requires a feature-space kernel, for which we use the radial basis function (RBF) kernel. In practice, such as on single-cell genomics data (Singh et al., 2020), MMD-based domain adaptation methods tend to struggle, for a few reasons. First, because MMD is a non-convex functional, it tends get stuck in local minima. This related to another practical weakness, which is that it is very sensitive to the choice of length-scale / bandwidth hyperparameter. When the bandwidth is too small, each datapoint are seen as dissimilar to all other points except itself. If the source and target data are separated, the second term in Eq. (4) will be approximately zero with vanishing gradient far from a skinny Gaussian, so no progress will be made. Yet when the bandwidth is too large, the gradient also vanishes with different datapoints together at the flat top of a wide Gaussian. Various measures have been proposed for these problems, such as adding a discriminative term to the objective (Wang et al., 2020) and choosing the (fixed) bandwidth in a data-driven way from the entire dataset (Singh et al., 2020).

We instead propose to dynamically recompute the bandwidth for each batch during the optimization procedure. Given source sample \( X^S \in \mathbb{R}^{N_{\text{batch}} \times M} \) and target sample \( X^T \in \mathbb{R}^{N_{\text{batch}} \times M} \) and the current transformation parameters \((A, b)\), the squared-bandwidth for a single dimension \( i \) of the features is computed as follows:

\[
\sigma_i^2 = \frac{1}{N_{\text{batch}}} \sum_{n=1}^{N_{\text{batch}}} \left( X^T_i - (AX^S_i + b_i) \right)^2.
\]

As our algorithm adapts the source to the target, our bandwidth estimate progressively adapts as well.

4.3 Estimators for the Conditional Distribution

In this section, we present three estimators for \( \mathcal{D}(\cdot | Y = y) \). The first two, multivariate Gaussian distribution and univariate Gaussian process, are designed to accompany the KL-divergence. The first allows one to model the joint distribution of all features but is homoscedastic. The second models each feature independently, so is only compatible with diagonal linear transformations, but is heteroscedastic. Furthermore, the first estimator is only able to model confounders with a linear effect on the features, with categorical features handled by one-hot encoding, while the latter models nonlinear effects with kernel \( k_{\mathcal{Y}}(y^{(n)}_1, y^{(n)}_1) \). For both the first two estimators, we ultimately estimate, conditioned on a given \( Y = y \), the mean and (co)variance of the conditional distribution. So both of these estimators are used in conjunction with the KL-divergence defined for two Gaussian distributions. Our third estimator, the Nadaraya-Watson model, requires no such distributional assumptions, at the cost of greater computational requirements.

4.3.1 Linear Gaussian Distribution for KL-divergence Minimization

Here we model \( x \) conditioned on real-valued vector \( y \) as linear Gaussian:

\[
\mathcal{D}(x|Y = y) = \mathcal{N}(By, \Lambda^{-1}).
\]
Because $x$ is potentially high-dimensional, we estimate parameters $(B, \Lambda)$ with regularized multivariate linear regression and the graphical lasso (Friedman et al., 2008). This model is homoscedastic, because all samples in the source dataset will have identical estimated covariances $\Sigma_S = \Lambda_S^{-1}$, $\Sigma_T = \Lambda_T^{-1}$. Meanwhile, given $N$ samples in the prior distribution, we will have different predicted means for source and target $\mu_S^{(n)}, \mu_T^{(n)}$, $\forall 1 \leq n \leq N$.

For the forward-KL divergence, this conditional distribution results in the following objective (after algebraic simplification):

$$
\min_{A, b} 2N \log(|A|) + \text{Tr} \left( A \Sigma_S A^\top \right)^{-1} \Sigma_T \\
+ \sum_{n=1}^{N} \left( A \mu_S^{(n)} + b - \mu_T^{(n)} \right)^\top \left( A \Sigma_S A^\top \right)^{-1} \left( A \mu_S^{(n)} + b - \mu_T^{(n)} \right).
$$

(10)

For the reverse-KL divergence, this conditional distribution results in the following:

$$
\min_{A, b} -2N \log(|A|) + \text{Tr} \left( \Sigma_T^{-1} A \Sigma_S A^\top \right) \\
+ \sum_{n=1}^{N} \left( A \mu_S^{(n)} + b - \mu_T^{(n)} \right)^\top \Sigma_T^{-1} \left( A \mu_S^{(n)} + b - \mu_T^{(n)} \right).
$$

(11)

Besides being easier to optimize, the reverse-KL objective minimizes the negative log-determinant of $A$, which functions as a log-barrier pushing it to have a positive determinant. This is useful, because the linear mapping between two Gaussians is not unique, and the reverse-KL divergence chooses the mapping which preserves rather than reverses the orientation.

That the $(- \log |A|)$ term arises naturally out of the reverse-KL divergence is of potential independent interest. Preventing collapse into trivial solutions is a known problem with MMD-based domain adaptation (Singh et al., 2020; Wu et al., 2021). The reverse-KL objective may inspire a new regularization penalty for this problem.

Furthermore, in the case of a location-scale adaptation, the reverse-KL divergence can be obtained via a fast exact closed-form solution. Further details are given in Appendix A.

4.3.2 Univariate Gaussian Process for KL-divergence Minimization

We will only use this model for location-scale transformations, so we model each feature independently. Without loss of generality, for feature $i$ in source domain $D_S(x|Y = y)$, we have $x_i \in \mathbb{R}^{N_S}$ modeled using a Gaussian process (GP):

$$f(y) \sim \mathcal{GP}(m(y), k_Y(y, y')).
$$

(12)

Having fit the GP on the source dataset $X_S^T \mapsto Y^S$, we evaluate it to compute and store the conditional mean and variance for each $y$ taken from the confounder prior. This process is repeated for all features, on both the source and target datasets.

4.3.3 Conditional Distribution Sampling for MMD Minimization

We model each of the conditional distributions $D_T(x|Y = y)$ and $D_S(x|Y = y)$ using Nadaraya-Watson kernel regression (Nadaraya, 1964; Watson, 1964). For each observed value of $Y = y$, we compute dataset sample weight $w(y^{(n)}) \propto k_Y(y, y^{(n)})$ for all samples in the target dataset and source dataset, respectively. Then, we sample (with repeats) from this distribution.

4.4 Choice of Confounder Distribution

Our choice of the confounder prior $D_Y^S$ should not matter greatly if our confounded shift assumption (which is noise-free) holds exactly, the true transformation is linear, and we have sufficient sample
size. If we recover the true transformation \(\hat{A} = A^*, \hat{b} = b^*\), then the distance/divergence will be 0 for all \(y \in \mathcal{Y}\). Nevertheless, because these assumptions are not expected to hold exactly in reality, the choice will matter. Based on our experience, we suggest setting \(D_{\mathcal{Y}}\) by default to be the empirical distribution of the concatenated source and target datasets \(\{Y_S, Y_T\}\).

**Additional Implementation Details** The design of confounder-space kernels is discussed in Appendix B. Computational speedups for categorical confounders are discussed in Appendix C.

### 5 Experiments

#### 5.1 Synthetic Data

##### 5.1.1 1d Data with 1d Continuous Confounder

We first examine confounded domain adaptation in the context of a single-dimensional feature confounded by a single-dimensional continuous confounder. Our results are illustrated in Figure 1. We analyze the performance of vanilla and ConDo adaptations, when the effect of the continuous confounder is linear homoscedastic (left column), linear heteroscedastic (middle column), and nonlinear heteroscedastic (right column). In all cases, there is confounded shift because in the target domain the confounder is uniformly distributed in \((0, 8)\), while for the source domain in \((4, 8)\).

We repeat the above experimental setup, but with modifications to verify whether our approach can be accurate even when its assumptions no longer apply. We run experiments with and without noisy batch effects, with-vs-without label shift (i.e. different distributions over the confounder between source and target), and with-vs-without feature shift (i.e. with and without batch effect), for a total of 8 shift-type settings.

Overall, we find that ConDo is robust to violations of the confounded shift assumption. We find that adding noise to the batch effect does not affect the performance of ConDo. We also find that without label shift (when confounding-awareness is unnecessary) ConDo is non-inferior to confounding-unaware methods. And, without feature-shift (when the true transformation is the identity, even if label shift makes the marginal feature distributions differ), we find that only ConDo reliably chooses approximately the identity transformation. The noise-free and noisy results are summarized in Tables 2 and 3, respectively. The full results are provided in the Appendix.

##### 5.1.2 1d Data with 1d Categorical Confounder

Here, we generate 1d features based on the value of a 1d categorical confounder. We also use this setting to analyze the performance of ConDo for a variety of sample sizes. For each sample size under consideration, we run 10 random simulations, and report the rMSE compared to the latent source domain values (before applying the target-to-source batch effect).

Results are shown in Figure 2. In Figure 2(A) we see that with a 200 source (and 200 target) samples, the ConDo methods have converged on the correct transformation, while their confounding-unaware analogues do not. We see in Figure 2(B) that with even 10 samples, our ConDo Linear Reverse-KL method correctly aligns the datasets. Meanwhile, with at least 100 samples, all our ConDo methods have smaller rMSE. Overall, we see that the non-MMD ConDo methods are robust to small sample sizes.

##### 5.1.3 Affine Transform for 2d Data with 1d Categorical Confounder

Next, we analyze the performance of our approach on 2d features requiring an affine (rather than location-scale) transformation. We also use this setting to assess the downstream performance of classifiers which are fed the adapted source-to-target features. The synthetic 2d features, before the batch effect, form a slanted “8” shape, shown in blue/green in Figure 3. Two linear classifiers, up-vs-down (in magenta) and left-vs-right (in cyan) are applied to this target domain.

Our results are shown in Figure 3. On the left column, we compare methods in the case where there is no confounded shift. (This setting is from Python Optimal Transport (Flamary et al., 2021).) In the middle column, we have induced a confounded shift: One-fourth of the source domain samples come from the upper loop of the “8”, while half of the target domain samples come from the upper loop.
Figure 1: ConDo methods are superior to Gaussian OT when confounded target shift and feature shift are present (with no noise added after shifting). Out of 9 (problem, ConDo method) combinations, ConDo methods do worse in only 1 combination (nonlinear heteroscedastic, linear Gaussian ReverseKL). The columns, in order, correspond to a confounder with a linear homoscedastic effect, a confounder with a linear heteroscedastic effect, and a confounder with a nonlinear heteroscedastic effect. The first row depicts the problem setup, while the remaining rows depict the performance of Gaussian OT and our ConDo methods. Red points overlapping with green points is indicative of high accuracy. In each subplot, we provide the rMSE on training source data (depicted), and in parentheses, the rMSE on heldout source data (not depicted) generated with confounder sampled from target prior $D_Y^T$. The printed rMSEs are averaged over 5 independent random simulation runs, while the plots depict the results from the final simulation run. Out of 9 (problem, ConDo method) combinations, ConDo does worse in only 1 combination (nonlinear heteroscedastic, linear Gaussian ReverseKL), which is expected since the linear Gaussian conditional distribution fails to model nonlinear effects and heteroscedasticity.
### Table 2: Summary of results for 1d data with 1d continuous confounder without noise

|                | Homoscedastic Linear | Heteroscedastic Linear | Nonlinear |
|----------------|----------------------|------------------------|-----------|
| **No Noise, Label-Shifted and Feature-Shifted** |                      |                        |           |
| Before Correction | 30.162 (24.211)      | 31.035 (24.724)        | 19.280 (14.036) |
| Oracle           | 0.000 (0.000)        | 0.000 (0.000)          | 0.000 (0.000) |
| Gaussian OT      | 9.184 (18.198)       | 8.347 (11.226)         | 5.804 (4.376)  |
| MMD              | 8.530 (6.581)        | 8.293 (6.270)          | 17.677 (12.830) |
| ConDo Linear-ReverseKL | 0.349 (0.740)      | 1.326 (1.476)          | 7.116 (7.289)  |
| ConDo GP-ReverseKL | 0.760 (1.494)       | 1.447 (1.883)          | 5.373 (2.760)  |
| ConDo MMD        | 2.310 (1.769)        | 4.373 (3.288)          | 5.032 (3.476)  |
| **No Noise, Label-Shifted Only**                |                      |                        |           |
| Before Correction | 0.000 (0.000)        | 0.000 (0.000)          | 0.000 (0.000) |
| Oracle           | 0.000 (0.000)        | 0.000 (0.000)          | 0.000 (0.000) |
| Gaussian OT      | 9.101 (17.144)       | 8.355 (10.435)         | 5.194 (4.280)  |
| MMD              | 6.667 (5.246)        | 8.820 (6.930)          | 0.000 (0.000)  |
| ConDo Linear-ReverseKL | 0.381 (0.494)      | 1.505 (1.760)          | 6.188 (6.947)  |
| ConDo GP-ReverseKL | 1.334 (2.051)       | 1.362 (1.693)          | 4.910 (4.249)  |
| ConDo MMD        | 2.377 (2.053)        | 4.800 (3.856)          | 0.000 (0.000)  |
| **No Noise, Feature-Shifted Only**               |                      |                        |           |
| Before Correction | 24.278 (22.620)      | 24.178 (24.612)        | 15.120 (13.531) |
| Oracle           | 0.000 (0.000)        | 0.000 (0.000)          | 0.000 (0.000) |
| Gaussian OT      | 0.903 (0.871)        | 0.796 (0.810)          | 1.651 (1.380)  |
| MMD              | 2.543 (2.342)        | 6.768 (6.636)          | 15.120 (13.531) |
| ConDo Linear-ReverseKL | 0.321 (0.330)      | 0.839 (0.825)          | 1.436 (1.222)  |
| ConDo GP-ReverseKL | 0.339 (0.353)       | 2.036 (2.031)          | 0.471 (0.449)  |
| ConDo MMD        | 1.180 (1.102)        | 1.192 (1.176)          | 2.533 (2.091)  |
| **No Noise, No Shift**                            |                      |                        |           |
| Before Correction | 0.000 (0.000)        | 0.000 (0.000)          | 0.000 (0.000) |
| Oracle           | 0.000 (0.000)        | 0.000 (0.000)          | 0.000 (0.000) |
| Gaussian OT      | 1.810 (1.881)        | 1.286 (1.261)          | 1.449 (1.446)  |
| MMD              | 2.411 (2.441)        | 2.340 (2.374)          | 0.544 (0.553)  |
| ConDo Linear-ReverseKL | 0.351 (0.357)      | 0.606 (0.617)          | 1.090 (1.094)  |
| ConDo GP-ReverseKL | 0.395 (0.392)       | 0.545 (0.538)          | 0.627 (0.641)  |
| ConDo MMD        | 0.825 (0.843)        | 0.641 (0.649)          | 0.267 (0.262)  |

This allows us to assess the affects of confounded shift on downstream prediction of the confounder (up-vs-down), as well as a non-confounder (left-vs-right). In the right column, we have induced a confounded shift as before, while making the true source-target transform more challenging, by having a non-negative element in the transformation matrix.

We see that ConDo Linear-ReverseKL is the only method that has small rMSE and high accuracy in all settings. All methods perform similarly where there is no confounded shift, but the vanilla domain adaptation approaches fail in the presence of confounding. Meanwhile, ConDo MMD has 25% training accuracy and 27% test accuracy on left-vs-right prediction problem in the challenging setting, because it flips the orientation of the data. This result highlights the benefits of the $(\log |A|)$ term in the reverse-KL divergence objective.

### 5.2 Real Data

We compare ConDo to baseline methods on image color adaptation and on gene expression batch correction.
Figure 2: Results for transforming 1d data with a 1d categorical confounder. (A) Depiction of original, batch-effected, and domain-adapted data, for each value of the categorical confounder. (B) Plot of rMSE vs sample size for each of the domain adaptation methods. Each rMSE was averaged over 10 simulations, with the vertical lines indicating 1 standard deviation over the simulations.

Figure 3: Results of affine transform of 2d data with a categorical confounder. We print the rMSE as well as the up-vs-down and left-vs-right accuracies on both the training data, and on heldout test data in parentheses. These values are the result of averaging over 5 random simulations, while the plot is generated from the final simulation.
Table 3: Summary of results for 1d data with 1d continuous confounder with noise

|                      | Noisy, Label-Shifted and Feature-Shifted | Noisy, Label-Shifted Only | Noisy, Feature-Shifted Only | Noisy, No Shift |
|----------------------|------------------------------------------|---------------------------|----------------------------|----------------|
|                      | Homoscedastic Linear | Heteroscedastic Linear | Nonlinear | Homoscedastic Linear | Heteroscedastic Linear | Nonlinear | Homoscedastic Linear | Heteroscedastic Linear | Nonlinear | Homoscedastic Linear | Heteroscedastic Linear | Nonlinear |
| Before Correction    | 30.373 (24.508) | 31.499 (23.668) | 19.415 (13.995) | 1.009 (0.975) | 0.971 (0.999) | 1.022 (1.036) | 23.511 (24.000) | 24.562 (24.529) | 14.667 (14.368) | 0.995 (1.026) | 1.017 (1.033) | 0.977 (1.000) |
| Oracle               | 0.517 (0.525) | 0.512 (0.514) | 0.499 (0.482) | 1.009 (0.975) | 0.971 (0.999) | 1.022 (1.036) | 4.909 (3.792) | 8.141 (6.528) | 1.022 (1.036) | 0.995 (1.026) | 1.017 (1.033) | 0.977 (1.000) |
| Gaussian OT          | 9.551 (17.369) | 8.737 (12.690) | 4.330 (3.779) | 9.767 (17.972) | 8.515 (10.887) | 4.394 (3.773) | 1.416 (1.484) | 1.817 (1.797) | 1.367 (1.296) | 1.750 (1.829) | 1.374 (1.330) | 1.871 (1.815) |
| MMD                  | 7.684 (6.015) | 10.019 (7.188) | 19.415 (13.995) | 2.271 (2.282) | 2.781 (2.904) | 14.667 (14.368) | 2.233 (2.144) | 1.976 (1.935) | 1.186 (1.133) | 2.233 (2.144) | 1.976 (1.935) | 1.186 (1.133) |
| ConDo Linear-ReverseKL | 0.626 (0.910) | 1.401 (1.746) | 6.782 (7.688) | 1.083 (1.030) | 1.588 (1.769) | 6.229 (6.922) | 0.586 (0.589) | 1.274 (1.289) | 1.511 (1.463) | 0.577 (0.580) | 1.276 (1.306) | 1.157 (1.167) |
| ConDo GP-ReverseKL    | 1.023 (1.809) | 2.113 (3.206) | 2.634 (2.043) | 1.181 (1.213) | 2.311 (1.996) | 3.906 (3.679) | 2.974 (2.485) | 4.863 (4.012) | 1.307 (1.194) | 1.150 (1.156) | 1.908 (1.861) | 2.670 (2.589) |
| ConDo MMD             | 2.919 (2.287) | 5.108 (3.651) | 4.223 (2.855) | 2.919 (2.287) | 5.108 (3.651) | 4.223 (2.855) | 2.919 (2.287) | 5.108 (3.651) | 4.223 (2.855) | 2.919 (2.287) | 5.108 (3.651) | 4.223 (2.855) |

5.2.1 Image Color Adaptation

We here apply domain adaptation to the problem of image color adaptation, depicted in Figure 4. We start by adapting back and forth between two ocean pictures taken during the daytime and sunset (from the Python Optimal Transport library (Flamary et al., 2021) Gaussian OT example). In this scenario, there is no confounding, since the images contains water and sky in equal proportions. Thus, conditioning on each pixel label (categorical, either “water” or “sky”), makes no difference, as expected. Next, we attempted color adaptation between the ocean daytime photo and another sunset photo including beach, water, and sky. Here, there is confounded shift, so ConDo successfully utilizes pixels labeled as “sky”, “water”, or “sand”. More results, including a depiction of pixel labelling, are in the Appendix.

5.2.2 Gene Expression Batch Effect Correction

We analyze performance on the bladderbatch gene expression dataset commonly used to benchmark batch correction methods (Leek, 2016). In our experiment, we attempt a location-scale transform, as is typical with gene expression batch effect correction. We choose the second largest batch (batch 2, with 4 cancer samples out of 18 total) as the source, and the largest batch (batch 5, with 5 cancer
samples out of 18 total) as the target. We use all 22,283 gene expressions in the dataset. In addition to the methods previously used, we also compare to ComBat ([Johnson et al., 2007]), the dominant method for removing batch effects in gene expression data. ComBat is an empirical Bayes method which learns a location-scale transformation by exploits the fact that large groups of genes should typically share (roughly) the same batch effect parameters. This allows it to perform well on small-$N$, high-dimensional data, which is common in gene expression studies. ComBat can also control for confounding variables which have a linear effect on gene expression. On the other hand, ComBat does not map a source domain to a target domain, but rather finds a shared latent domain that both domains may be linearly mapped to.

Because the cancer fractions are roughly the same for batches 2 and 5, we do not expect to need to account for confounding. Results are shown in Figure 5(A). For each method, we visualize the effects of correction with t-SNE ([Van der Maaten & Hinton, 2008]) and PCA. We see that all methods are roughly equally successful at mixing together the samples from different batches (i.e., by color), while keeping cancer vs not-cancer samples clustered apart (i.e., X versus O). For each method, we also compute the silhouette scores of the adapted datasets, with respect to the batch variable (and, in parentheses, the test result variable). We desire the silhouette score to be small for the batch variable, and big for the test result variable.

We repeat the experiment after removing half (7) of the non-cancer samples in batch 2, so that batch 2 is 4/11 non-cancerous, while batch 5 remains 5/18 non-cancerous. Results are shown in Figure 5(B). We see that ConDo linear Gaussian method performs better than vanilla Gaussian OT, and ConDo MMD performs better than vanilla MMD. However, ConDo is still not as good as ComBat, which is able to perform separate transforms on both batches, and shares information across genes.
Figure 5: Results on bladderbatch dataset, without confounded shift (A), and with confounded shift (B). We would like reds and blues to be well-mixed, while cancer and non-cancer samples to cluster apart. In both (A) and (B), we show t-SNE on the left, and PCA on the right.
6 Conclusion and Future Work

6.1 Conclusion

We have shown that minimizing expected divergences / distances after conditioning on confounders is a promising avenue for domain adaptation in the presence of confounded shift. Our proposed use of the reverse KL-divergence and our dynamic choice of RBF kernel bandwidth are (to our knowledge) new in the field of domain adaptation, and may be more broadly useful. Focusing on settings where the effect of the confounder is possibly complex, yet where the source-target domains can be linearly adapted, we demonstrated the usefulness of both parametric and nonparametric algorithms based on our framework. Our ConDo framework seems to learn adaptations that are good for a variety of downstream tasks, including prediction and clustering.

6.2 Future Work

Our approach is more appropriate for adaptation settings where source and target correspond to different versions of sensor devices, different laboratory protocols, and similar settings where the required adaptation is linear (or even location-scale). It would be useful to examine whether this approach extends gracefully to nonlinear adaptations, such as those parameterized by neural networks.

Our KL-divergence based approach is currently relies on either a (potentially multivariate) linear Gaussian distribution or a univariate (nonlinear) Gaussian Process (GP). Extending the latter to full affine transformations of multivariate features could take advantage of recent advances in using Gaussian Process conditional density estimation (Dutordoir et al. 2018) for better modeling of uncertainty, and recent advances in improving scalability for multivariate outputs (Zhe et al. 2019).

Optimization of MMD is challenging, because it is a nonconvex functional. Our sampling from the confounder prior injects noise which may help overcome the nonconvexity, but adding Gaussian noise to the samples has been proven to be beneficial (Arbel et al. 2019), so it is worth examining.

Furthermore, while we minimized the KL-divergence and MMD, there are other potential distances worth minimizing. For example, Wasserstein Procrustes analysis was recently developed and applied to align text embeddings across languages (Grave et al. 2019; Ramírez et al. 2020). By combining this with conditioning on confounding variables, one could potentially align embeddings between languages with different topic compositions.

Finally, thus far our analysis of ConDo has been purely empirical. Theoretical analysis would surely be appropriate, particularly before applying it to data analyses and statistical inference tasks.
References

Arbel, Michael, Korba, Anna, Salim, Adil, & Gretton, Arthur. 2019. Maximum mean discrepancy gradient flow. Advances in Neural Information Processing Systems, 32.

Blei, David M, Kucukelbir, Alp, & McAuliffe, Jon D. 2017. Variational inference: A review for statisticians. Journal of the American statistical Association, 112(518), 859–877.

Dowson, DC, & Landau, BV. 1982. The Fréchet distance between multivariate normal distributions. Journal of multivariate analysis, 12(3), 450–455.

Dutordoir, Vincent, Salimbeni, Hugh, Hensman, James, & Deisenroth, Marc. 2018. Gaussian process conditional density estimation. Advances in neural information processing systems, 31.

Flamary, Rémi, Courty, Nicolas, Gramfort, Alexandre, Alaya, Mokhtar Z., Boisbunon, Aurélie, Chambron, Stanislas, Chapel, Laetitia, Corenflos, Adrien, Fatras, Kilian, Fournier, Nemo, Gautheron, Léo, Gayraud, Nathalie T.H., Janati, Hicham, Rakotomamonjy, Alain, Redko, Ievgen, Rolet, Antoine, Schutz, Anthony, Seguy, Vivien, Sutherland, Danica J., Tavenard, Romain, Tong, Alexander, & Vayer, Titouan. 2021. POT: Python Optimal Transport. Journal of Machine Learning Research, 22(78), 1–8.

Friedman, Jerome, Hastie, Trevor, & Tibshirani, Robert. 2008. Sparse inverse covariance estimation with the graphical lasso. Biostatistics, 9(3), 432–441.

Grave, Edouard, Joulin, Armand, & Berthet, Quentin. 2019. Unsupervised alignment of embeddings with Wasserstein Procrustes. Pages 1880–1890 of: The 22nd International Conference on Artificial Intelligence and Statistics. PMLR.

Johnson, W Evan, Li, Cheng, & Rabinovic, Ariel. 2007. Adjusting batch effects in microarray expression data using empirical Bayes methods. Biostatistics, 8(1), 118–127.

Kappen, Hilbert J, Gómez, Vicenç, & Opper, Manfred. 2012. Optimal control as a graphical model inference problem. Machine learning, 87(2), 159–182.

Knott, Martin, & Smith, Cyril S. 1984. On the optimal mapping of distributions. Journal of Optimization Theory and Applications, 43(1), 39–49.

Leek, JT. 2016. bladderbatch: Bladder gene expression data illustrating batch effects. R package version, 1(0).

Levine, Sergey. 2018. Reinforcement learning and control as probabilistic inference: Tutorial and review. arXiv preprint arXiv:1805.00909.

MacQueen, James. 1967. Some methods for classification and analysis of multivariate observations. Pages 281–297 of: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, vol. 1. Oakland, CA, USA.

Mallasto, Anton, & Feragen, Aasa. 2017. Learning from uncertain curves: The 2-Wasserstein metric for Gaussian processes. Advances in Neural Information Processing Systems, 30.

Muzellec, Boris, & Cuturi, Marco. 2018. Generalizing point embeddings using the Wasserstein space of elliptical distributions. Advances in Neural Information Processing Systems, 31.

Nadaraya, Elizbar A. 1964. On estimating regression. Theory of Probability & Its Applications, 9(1), 141–142.

Peyré, Gabriel, Cuturi, Marco, et al. 2019. Computational optimal transport: With applications to data science. Foundations and Trends® in Machine Learning, 11(5-6), 355–607.

Ramírez, Guillelm, Dangovski, Rumen, Nakov, Preslav, & Soljačić, Marin. 2020. On a Novel Application of Wasserstein-Procrustes for Unsupervised Cross-Lingual Learning. arXiv preprint arXiv:2007.09456.

Shafieezadeh Abadeh, Soroosh, Nguyen, Viet Anh, Kuhn, Daniel, & Mohajerin Esfahani, Peyman M. 2018. Wasserstein distributionally robust Kalman filtering. Advances in Neural Information Processing Systems, 31.
Singh, Ritambhara, Demetci, Pinar, Bonora, Giancarlo, Ramani, Vijay, Lee, Choli, Fang, He, Duan, Zhijun, Deng, Xinxian, Shendure, Jay, Distèche, Christine, et al. 2020. Unsupervised manifold alignment for single-cell multi-omics data. Pages 1–10 of: Proceedings of the 11th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics.

Tachet des Combes, Remi, Zhao, Han, Wang, Yu-Xiang, & Gordon, Geoffrey J. 2020. Domain adaptation with conditional distribution matching and generalized label shift. Advances in Neural Information Processing Systems, 33, 19276–19289.

Van der Maaten, Laurens, & Hinton, Geoffrey. 2008. Visualizing data using t-SNE. Journal of machine learning research, 9(11).

Wang, Wei, Li, Haojie, Ding, Zhengming, & Wang, Zhihui. 2020. Rethink maximum mean discrepancy for domain adaptation. arXiv preprint arXiv:2007.00689.

Watson, Geoffrey S. 1964. Smooth regression analysis. Sankhyā: The Indian Journal of Statistics, Series A, 359–372.

Wu, Xiaofu, Zhang, Suofei, Zhou, Quan, Yang, Zhen, Zhao, Chunming, & Latecki, Longin Jan. 2021. Entropy Minimization Versus Diversity Maximization for Domain Adaptation. IEEE Transactions on Neural Networks and Learning Systems.

Zhang, Kun, Schölkopf, Bernhard, Muandet, Krikamol, & Wang, Zhiyuan. 2013. Domain adaptation under target and conditional shift. Pages 819–827 of: International Conference on Machine Learning. PMLR.

Zhe, Shandian, Xing, Wei, & Kirby, Robert M. 2019. Scalable high-order gaussian process regression. Pages 2611–2620 of: The 22nd International Conference on Artificial Intelligence and Statistics. PMLR.
A Exact Solution for 1d Reverse KL-divergence

For each $n$th sample ($1 \leq n \leq N$) drawn from the prior distribution over the confounding variable, we have obtained an estimate of its mean and variance for the source domain $(\mu_{S_n}, \sigma_{S_n}^2)$ and target domain $(\mu_{T_n}, \sigma_{T_n}^2)$. Then the reverse-KL objective is the following:

$$\arg \min_{m,b} \sum_{n=1}^{N} - \log(m) + \frac{m^2 \sigma_{S_n}^2}{2 \sigma_{T_n}^2} + \frac{(m \mu_{S_n} + b - \mu_{T_n})^2}{2 \sigma_{T_n}^2}$$

(13)

$$= \arg \min_{m,b} \sum_{n=1}^{N} - \log(m)(2 \sigma_{T_n}^2) + m^2 \sigma_{S_n}^2 + (m \mu_{S_n} + b - \mu_{T_n})^2.$$  

(14)

Setting the partial derivative wrt $b$ to 0, we have:

$$b^* = \frac{1}{N} \left( \sum_{n=1}^{N} \mu_{T_n} - m \left( \sum_{n=1}^{N} \mu_{S_n} \right) \right) = \mu_T - m \mu_S.$$  

(15)

Substituting this into our objective, we have

$$\arg \min_{m} \sum_{n=1}^{N} - \log(m)(2 \sigma_{T_n}^2) + m^2 \sigma_{S_n}^2 + (m \mu_{S_n} + \mu_T - m \mu_S - \mu_{T_n})^2.$$  

(16)

Setting the derivative wrt $m$ to 0, we obtain the following quadratic equation:

$$\left[ \left( \sum 2 \sigma_{S_n}^2 \right) + \left( \sum 2(\mu_{S_n} - \mu_S)^2 \right) \right] m^2 + \left[ \sum 2(\mu_T - \mu_{T_n})(\mu_{S_n} - \mu_S) \right] m + \left[ -2 \sum \sigma_{T_n}^2 \right].$$  

(17)

We then apply the quadratic formula, choosing the positive solution.

B Design of Confounder-Space Kernels

Both the MMD distance and GP estimator require a confounder-space kernel. For each quantitative confounder, the kernel function is itself the sum of an RBF kernel and a zero-mean heteroscedastic kernel. The heteroscedastic kernel predicts noise levels via kernel regression with 10 prototypes, where the 10 prototypes are found by KMeans clustering (MacQueen, 1967). For each categorical confounder, the kernel function is itself the sum of a white kernel (with a background similarity of $10^{-8}$) and a zero-mean heteroscedastic kernel.

C Computational Speedups for Categorical Confounders

If $Y$ is a categorical variable with cardinality less than the sample size, we speed up performance by drawing from unique values of $Y$, and weighting each unique value by its count.

For MMD, this reduces the complexity of processing each batch from $O(N N_{batch}^2 M)$ to $O(\tilde{N} N_{batch}^2 M)$, where $\tilde{N} \leq N$ is the number of unique samples in our prior distribution dataset. (If we sample proportionally from the source and target, $\tilde{N} = N_S + N_T$, so this is a substantial speedup.) For the GP conditional distribution estimators, this reduces the kernel-computation costs for source and target domains from $O(N N_S^2)$ and $O(N N_T^2)$ to $O(\tilde{N} N_S^2)$ and $O(\tilde{N} N_T^2)$, respectively.

D Additional Material for the Experiments

D.1 1d Data with 1d Continuous Confounder

The remaining noise-free results are provided in Figures S6, S5, and S7. The results for the noise batch effect settings are in Figures S1, S3, S2, and S4.
Figure S1: ConDo methods substantially improve on Gaussian OT in the presence of confounded shift. In this experiment, both label shift and covariate shift are present and confounded with each other, and \( \mathcal{N}(0, 1) \) noise is added to the source data after the shifting. (See the caption of Figure 1 for how to read this figure.)
Figure S2: ConDo methods are still superior to Gaussian OT when target shift, but not feature shift is present (and when $N(0, 1)$ noise is added to the source data after the shifting). Out of 9 (problem, ConDo method) combinations, ConDo methods are only worse in 1 combination: (nonlinear heteroscedastic, linear Gaussian ReverseKL). In this scenario, the best source-to-target transform is simply the identity; we see that the ReverseKL methods learn to do this when the confounder’s effect is linear, while the MMD methods learn to do this when the confounder’s effect is nonlinear. (See the caption of Figure 1 for how to read this figure.)
Figure S3: ConDo methods are generally superior to Gaussian OT when feature shift, but not target shift is present (and when $\mathcal{N}(0, 1)$ noise is added to the source data after the shifting). Out of 9 (problem, ConDo method) combinations, ConDo methods are only worse in 3 combinations: (nonlinear heteroscedastic, Linear Gaussian ReverseKL), (linear homoscedastic, MMD), and (nonlinear heteroscedastic, MMD). (See the caption of Figure 1 for how to read this figure.)
Figure S4: ConDo methods are still superior to Gaussian OT when neither target shift nor feature shift is present (and when $\mathcal{N}(0,1)$ noise is added to the source data after the shifting). There is no setting where any of the ConDo methods is worse than Gaussian OT. In this scenario, the best source-to-target transform is simply the identity; we see that the ConDo methods are generally able to learn to do nothing. (See the caption of Figure 1 for how to read this figure.)
Figure S5: ConDo methods are generally superior to Gaussian OT when target shift, but not feature shift is present (with no noise added after shifting). Out of 9 (problem, ConDo method) combinations, ConDo methods do worse in only 1 combination (nonlinear heteroscedastic, linear Gaussian ReverseKL). (See the caption of Figure S1 for how to read this figure.)
Figure S6: ConDo methods behave similarly well as Gaussian OT when feature shift, but not target shift is present (with no noise added after shifting). Out of 9 (problem, ConDo method) combinations, ConDo methods do better in 5 combinations, but worse in 4 combinations. The ReverseKL methods generally do better than Gaussian OT, while MMD struggles in this case. (See the caption of Figure 1 for how to read this figure.)
Figure S7: ConDo methods are still superior to Gaussian OT when neither target shift nor feature shift is present (with no noise added after shifting). There is no setting where any of the ConDo methods is worse than Gaussian OT. In this scenario, the best source-to-target transform is simply the identity; we see that the ConDo methods are generally able to learn to do nothing. (See the caption of Figure 1 for how to read this figure.)
D.2 1d Data with 1d Categorical Confounder

Figure S8 provides additional results for our experiments with a 1d feature confounded by a 1d categorical confounding variable. By visual inspection of Figure S8(A) we see that, even with 10 samples, our ConDo Linear Reverse-KL method aligns the data correctly. Figure S8(E) shows that the ConDo MMD method has relatively worse scalability in terms of sample size, compared to the non-ConDo and other ConDo methods.

Figure S8: Additional results for 1d feature confounded by 1d category. Results are shown (A) \( N = 10 \), (B) \( N = 20 \), (C) \( N = 50 \), and (D) \( N = 100 \) samples. Subplot (E) shows runtime in seconds as a function of sample size.

D.3 2d Data with 1d Categorical Confounder

The procedure for generating data in this scenario is adapted from the Python Optimal Transport library [Flamary et al. 2021]. The batch-effected source data are generated first, with \( N = 200 \) split between two circles centered at \((0,0)\) and \((0,2)\); the points are distributed with angle distributed iid around the circle from \( U[0, 2\pi] \), and with radius sampled iid from \( \mathcal{N}(0, 1) \). The target domain data are generated via affine transform

\[
[0.1]
\tag{18}
\]

D.4 Image Color Adaptation

MMD is run for \( 10^{-3} \) epochs, with a batch size of 128, learning rate of \( 10^{-3} \), and momentum of 0.9. ConDo-MMD is run for \( 10^{-4} \) epochs, with a batch size of 128, learning rate of \( 10^{-8} \), and momentum of 0.9.
Figure S9: (A) Image color adaptation results with confounded shift, from sunset to beach sunset. (B) Pixel labels for beach sunset photo.

In Figure S9(A) we see that ConDo Linear is better than (linear) Gaussian OT. On the other hand, ConDo MMD is worse than vanilla MMD on the inverse mapping, because we are unable to prevent a change of handedness which leads poor inverse mappings. In Figure S9(B) we show the pixel labelling (i.e. the confounding variable value for each pixel) used by ConDo methods.

D.5 Gene Expression Batch Correction

Both MMD and ConDo-MMD are run for 10 epochs, with batch size 16, and momentum parameter 0.999. MMD uses a learning rate of $10^{-3}$, while ConDo-MMD uses a learning rate of $10^{-4}$. 