RealCause: Realistic Causal Inference Benchmarking

Brady Neal [1] **Chin-Wei Huang** [1] **Sunand Raghupathi** [1]

Abstract

There are many different causal effect estimators in causal inference. However, it is unclear how to choose between these estimators because there is no ground-truth for causal effects. A commonly used option is to simulate synthetic data, where the ground-truth is known. However, the best causal estimators on synthetic data are unlikely to be the best causal estimators on realistic data. An ideal benchmark for causal estimators would both (a) yield ground-truth values of the causal effects and (b) be representative of real data. Using flexible generative models, we provide a benchmark that both yields ground-truth and is realistic. Using this benchmark, we evaluate 66 different causal estimators.

1. Introduction

In causal inference, we want to measure causal effects of treatments on outcomes. Given some outcome Y and a binary treatment T, we are interested in the *potential outcomes* $Y_i(1)$ and $Y_i(0)$. Respectively, these denote the outcome that unit i would have if they were to take the treatment ($T = 1$) and the outcome they would have if they were to not take the treatment ($T = 0$). We are often interested in causal estimands such as $\mathbb{E}[Y(1)] - \mathbb{E}[Y(0)]$, the *average treatment effect* (ATE). This is equivalent to the following expression using Pearl's do-notation (Pearl, 1994; 2009; 2019): $\mathbb{E}[Y \mid \text{do}(T = 1)] - \mathbb{E}[Y \mid \text{do}(T = 0)]$, where $\text{do}(T = t)$ is a more mnemonic way of writing that we set the value of the treatment to t.

There are many different estimators for estimating causal estimands (see, e.g., Neal, 2020; Hernán & Robins, 2020; Morgan & Winship, 2014; Imbens & Rubin, 2015, and Section 6). However, it is unclear how to choose between these estimators because the true values of the causal estimands are generally unknown. This is because we cannot observe both potential outcomes (Rubin, 1974), so we have no ground-truth. This is often referred to as the *fundamental problem of causal inference* (Holland, 1986). Supervised machine learning does not have this "no ground-truth" problem because it is only interested in estimating $\mathbb{E}[Y \mid T]$, which only requires samples from $P(Y \mid T)$, rather than samples from $P(Y \mid \text{do}(T = 1))$ and $P(Y \mid \text{do}(T = 0))$. Yet, we must choose between causal estimators. How can we do that when faced with the fundamental problem of causal inference?

To evaluate causal estimators, people have created various benchmarks, each bringing different strengths and weaknesses that we will cover in Section 3. In this paper, we focus on how well causal estimators perform in the simplest setting, where there is no unobserved confounding, no selection bias, and no measurement error. The ideal benchmark for choosing between causal estimators in this setting should have the following qualities:

1. yield ground-truth estimands
2. be representative of a substantial subset of real data
3. all of the confounders are observed
4. yield many different data distributions of varying important characteristics (e.g. degree of overlap)

Item 1 is important in order to know which estimators yield estimates closer to the ground-truth. Item 2 is important so that we know that estimators that perform well on our benchmark will actually perform well on real datasets that we would apply them to. Item 3 is important so that we can rule out unobserved confounding as the explanation for an estimator performing poorly. Item 4 is important because it is unlikely that rankings of causal estimators on a single problem will generalize perfectly to all problems. Rather, we might expect that certain estimators perform better on distributions that have certain properties and other estimators perform better on distributions that have other specific properties. Existing benchmarks often have 1-3 of the above qualities (Section 3). Our benchmarking framework has all four.

We present a benchmark that simulates data from data generating processes (DGPs) that are statistically indistinguishable from observed real data. We first take the observed pretreatment covariates W as the only common causes of

[1]Mila, Université de Montréal. Correspondence to: Brady Neal <bradyneal11@gmail.com>.

Working paper.

T and Y. Then, we fit generative models $P_{\text{model}}(T \mid W)$ and $P_{\text{model}}(Y \mid T, W)$ that closely match the real analogs $P(T \mid W)$ and $P(Y \mid T, W)$. This allows us to simulate realistic data by first sampling W from the real data, then sampling T from $P_{\text{model}}(T \mid W)$, and finally sampling Y from $P_{\text{model}}(Y \mid T, W)$. Importantly, because we've fit generative models to the data, we can sample from *both* interventional distributions $P_{\text{model}}(Y \mid T = 1, W)$ and $P_{\text{model}}(Y \mid T = 0, W)$, which means that we have access to ground-truth estimands for our realistic simulated data. In other words, the fundamental problem of causal inference isn't a problem in these DGPs. We then use this realistic simulated data for benchmarking.

2. Preliminaries and Notation

We use upper-case letters to denote random variables and use lower-case letters to denote specific non-random values (except in the case of unit-level potential outcomes, which are not random). Let T be a binary scalar random variable denoting the treatment. Let W be a set of random variables that corresponds to the observed covariates. Let Y be a scalar random variable denoting the outcome of interest. Let $e(w)$ denote the *propensity score* $P(T = 1|W = w)$. We denote the treatment and outcome for unit i as T_i and Y_i. $Y_i(1)$ denotes the potential outcome that unit i would observe if T_i were 1 (takes treatment), and $Y_i(0)$ denotes the potential outcome that unit i would observe if T_i were 0 (does not take treatment). $Y(t)$ is a random variable that is a function of all the relevant characteristics I (a set of random variables) that characterize the outcome of an individual (unit) under treatment t. Concretely,

$$Y(t) = f(t, I)\,, \tag{1}$$

for some deterministic function f. Therefore, $Y_i(t)$ is deterministic. It is completely determined by the specific characteristics i for unit i and the treatment t.

We define the *individual treatment effect* (ITE) for unit i as:

$$\tau_i \triangleq Y_i(1) - Y_i(0)\,. \tag{2}$$

We define the *average treatment effect* (ATE) as

$$\tau \triangleq \mathbb{E}[Y(1) - Y(0)] \tag{3}$$

Let C be a set of random variables, denoting all the common causes (confounders) of the causal effect of T on Y. We can identify the ATE from observational data if we observe C. This setting has many names: "no unobserved confounding," "conditional ignorability," "conditional exchangeability," 'selection on observables," etc. In this setting, we can identify the ATE via the *adjustment formula* (Robins, 1986; Spirtes et al., 1993; Pearl et al., 2016; Pearl, 2009):

$$\tau = \mathbb{E}_C\left[\mathbb{E}[Y \mid T = 1, C] - \mathbb{E}[Y \mid T = 0, C]\right] \tag{4}$$

We define the *conditional average treatment effect* (CATE) similarly:

$$\tau(x) \triangleq \mathbb{E}[Y(1) - Y(0) \mid X = x] \tag{5}$$

$$= \mathbb{E}_C\left[\mathbb{E}[Y|T = 1, x, C] - \mathbb{E}[Y|T = 0, x, C]\right] \tag{6}$$

Here, X is a set of random variables that corresponds to the characteristics that we are interested in measuring more specialized treatment effects with respect to (x-specific treatment effects). This is a finer treatment effect that the more coarse ATE.

If $I \subseteq X$,[1] then there is an equivalence between ITEs and CATEs (assuming there are no colliders in X that open up backdoor paths): $\tau(x_i) = \tau_i$. If not, we can still have a treatment effect that is as specialized as possible, given the observed covariates W. This is simply the CATE where $X = W$, so we denote it by $\tau(w)$. Following Knaus et al. (2018), we call $\tau(w)$ the *individualized average treatment effect* (IATE). It is not uncommon to see people refer to "IATEs" as "ITEs." We will focus on IATEs in this paper.

In this minimum viable paper, we consider DGPs where $W = C$, for simplicity. This means that we must adjust for all of W to get causal effects and that the IATEs reduce to

$$\tau(w) = \mathbb{E}[Y|T = 1, w] - \mathbb{E}[Y|T = 0, w] \tag{7}$$

$$\triangleq \mu(1, w) - \mu(0, w)\,, \tag{8}$$

where μ is the *mean conditional outcome*. Our DGPs provide ground-truth IATEs by providing μ. They could provide ground-truth ITEs by providing f (Equation 1), but this would force them to project Y onto W. Just providing μ allows our DGPs to capture unobserved causes of Y in the data. Such causes are completely reasonable to expect in the real data if $I \not\subseteq W$. We will consider DGPs where the observed covariates are all confounders ($W = C$).

3. Methods for Evaluating Causal Estimators

3.1. Simulated Synthetic Data

The simplest way to get ground truth ATEs is to simulate synthetic data that we construct so that the only confounders of the effect of T on Y are W. This gives us access to the true *outcome mechanism* $P(Y \mid T, W)$. Using the outcome mechanism, we have access to the ground-truth IATE via Equation 7 and the ground-truth ATE via Equation 4.

In these simulations, we additionally have access to the true *treatment selection mechanism* $P(T \mid W)$ (or just "*selection mechanism*" for short). We must be able to sample from this to generate samples from $P(W, T, Y)$ through ancestral sampling:

$$P(W) \to P(T \mid W) \to P(Y \mid T, W) \tag{9}$$

[1] In measure-theoretic terms, if I is measurable with respect to the σ-algebra generated by X

Having access to $P(T \mid W)$ gives us ground-truth for things like the propensity scores and the degree of positivity/overlap violations.

This is probably the most common method for evaluating causal estimators. However, it has several disadvantages. First, the data is completely synthetic, so we do not know if the rankings of estimators that we get will actually generalize to real data. Second, authors proposing new causal estimators are naturally interested in synthetic data with specific properties that their estimator was developed to perform well on. This means that different synthetic data is used in different papers, so it is often not helpful for fairly comparing estimators.

3.2. Simulated Semi-Synthetic Data with Real Covariates

One natural improvement on the completely synthetic data described in Section 3.1 is to make it more realistic by taking the covariates W from real data. This means that $P(W)$ is realistic. Then, one can proceed with generating samples through Equation 9 by simulating $P(T \mid W)$ and $P(Y \mid T, W)$ as arbitrary stochastic functions. One of the main advantages of this is that these stochastic functions can be made to have any properties that its designers choose such as degree of nonlinearity, degree of positivity violation, degree of treatment effect heterogenity, etc. (Dorie et al., 2019).

This is what many current benchmarks do (Dorie et al., 2019; Shimoni et al., 2018; Hahn et al., 2019). The main problem is that the selection mechanism $P(T \mid W)$ and outcome mechanism $P(Y \mid T, W)$ are unrealistic.

3.3. Simulated Data that is Fit to Real Data

The way to fix the unrealistic selection and outcome mechanisms that we discussed in Section 3.2 is fit them to real data. This is what we do, and we are not the first. For example, there is work on this in economics (Knaus et al., 2018; Athey et al., 2019; Huber et al., 2013; Lechner & Wunsch, 2013), in healthcare (Wendling et al., 2018; Franklin et al., 2014), and in papers that are meant for a general audience (Abadie & Imbens, 2011; Schuler et al., 2017). Some fit relatively simple models (Franklin et al., 2014; Abadie & Imbens, 2011), whereas others fit more flexible models (Wendling et al., 2018; Athey et al., 2019; Schuler et al., 2017).

Our work is distinguished from the above work in two ways:

First, most of the above work does not use modern generative models that can fit most distributions so well that they pass two-sample tests.[2] Athey et al. (2019) is the exception as they use conditional Wasserstein Generative Adversarial Networks (WGANs) (Goodfellow et al., 2014; Arjovsky et al., 2017) for both the outcome mechanism and the reverse of the selection mechanism: $P(W \mid T)$. However, they do not report two-sample tests to rigorously test the hypothesis that their generative model is statistically similar to the distributions they are fit to. We fit several datasets well and run two-sample tests to test this claim in Section 5.

Second, our method allows us to have "knobs" to vary important aspects of the data distributions, just as Dorie et al. (2019) are able to maintain in their semi-synthetic study where they specify random functions for the outcome mechanism and selection mechanism. Wendling et al. (2018) illustrate the nontriviality of this when they wrote "The design of a simulation study is usually a trade-off between realism and control." We are able to get both realism *and* control.

3.4. Using RCTs for Ground-Truth

Constructed observational studies One can first take a randomized control trial (RCT), and get an unbiased estimate of the ground-truth ATE from that. Then, one can construct a corresponding observational study by swapping out the RCT control group with observational data for people who were not part of the RCT control group. LaLonde (1986) was the first to do this. We refer to this type of study as a *constructed observational study* (a term coined by Hill et al. (2004)). There are two problems with this type of study: (1) We do not know if we have observed all of the confounders for the observational data, so we do not know if an estimate that differs from the RCT ATE is due to unobserved confounding or due to the estimator doing poorly regardless of unobserved confounding. (2) The population that the observational data comes from is often not the same population as the population that the RCT data comes from, so it is not clear if the RCT ATE is the same as the true ATE of the constructed observational data.

Doubly randomized preference trials (DRPTs) In a *double randomized preference trial* (DRPT), one runs an RCT and an observational study in parallel on the same population. This can be done by first randomizing units into the RCT or observational study. The units in the RCT are then randomized into treatment groups. The units in the observational study are allowed to select which treatment they take. Shadish et al. (2008) were the first to run a DRPT to evaluate observational methods. The main problem with DRPTs (which is also a problem for constructed observational studies) is that you only get a single DGP, and it is prohibitively expensive to run many different DRPTs, which is important because we do not expect that the rankings of estimators will be the same across all DGPs. Additionally,

[2] In related work outside of causal inference, Turner & Neal (2018) applied modern generative models and two-sample tests for benchmarking Markov chain Monte Carlo (MCMC) samplers.

if a causal estimator performs poorly, we do not know if it is simply because of unobserved confounding.

Introducing selection bias via selective subsampling While $\hat{\mathbb{E}}[Y \mid T = 1]$ - $\hat{\mathbb{E}}[Y \mid T = 0]$ is an unbiased estimate of the ATE in an RCT, we can turn this into a biased estimate if we introduce selection bias (see, e.g., Kallus et al. (2018)). We can introduce selection bias by selectively subsampling the data based on T and Y and giving that subsampled data to the causal estimator. Graphically, this creates a collider C that is a child of both T and Y in the causal graph. And we're conditioning on this collider by giving the estimator acccess to only the subsampled data. This introduces selection bias (see, e.g., Hernán & Robins (2020, Chapter 8)), which means that $\hat{\mathbb{E}}[Y \mid T = 1]$ - $\hat{\mathbb{E}}[Y \mid T = 0]$ is a biased estimate in the subsampled data. The two problems with this approach are (a) the selection mechanism is chosen by humans, so it may not be realistic, and (b) the graphical structural of selection bias is different from the graphical structure of confounding (common effect of T and Y vs. common cause of T and Y).

4. RealCause: A Method for Producing Realistic Benchmark Datasets

The basic idea is to fit flexible generative models $P_{\text{model}}(T \mid W)$ and $P_{\text{model}}(Y \mid T, W)$ to the selection mechanism $P(T \mid W)$ and the outcome mechanism $P(Y \mid T, W)$, respectively. For $P_{\text{model}}(W)$, we simply sample from $P(W)$, just as was done in the semi-synthetic data simulations we described in Section 3.2. These three mechanisms give us a joint $P_{\text{model}}(W, T, Y)$ that we would like to be the same as the true $P(W, T, Y)$. This is what makes our DGPs realistic.

Architecture We use neural networks to parameterize the conditioning of $P_{\text{model}}(T \mid W)$ and $P_{\text{model}}(Y \mid T, W)$; that is the input of the neural net is either W (to predict the selection) or both W and T (to predict the outcome). A naive approach would be to concatenate W and T to predict the outcome, but our preliminary experiments on semi-synthetic data (where the true ATE is known) suggest that the resulting generative model tends to underestimate the true ATE. For example, this can happen from the network "ignoring" T, especially when W is high-dimensional. Therefore, we follow the TARNet structure (Shalit et al., 2017) to learn two separate conditionals $P_{\text{model}}(Y \mid T = 0, W)$ and $P_{\text{model}}(Y \mid T = 1, W)$, encoding the importance of T into the structure of our network. Since all conditionals depend on W, we use a multi-layer perceptron (MLP) to extract common features $h(W)$ of W. We then have three more MLPs to model T, $Y \mid T = 0$, and $Y \mid T = 1$ separately, taking in the features $h(W)$ as input. These all use the same $h(W)$, which is also learned, like in Dragonnet (Shi et al., 2019).

For simplicity, all four MLPs have the same architecture (except for the output dimensionality), with the tunable hyperparameters being the number of layers, the number of hidden units, and the activation function.

Distribution Assumption We use the output of the MLPs to parameterize the distributions of selection and outcome. For example, for binary data (such as treatment), we apply the logistic sigmoid activation function to the last layer to parameterize the mean parameter of the Bernoulli distribution. For real-valued data (such as the outcome variable), one option is to assume it follows a Gaussian distribution conditioned on the covariates, in which case we would have the neural net output the mean and log-variance parameters. The baseline model that we use is a linear model that outputs the parameters of a Gaussian distribution with a diagonal covariance matrix. The main (more flexible) generative model we use is the sigmoidal flow (Huang et al., 2018), which has been shown to be a universal density model capable of fitting arbitrary distributions.

For mixed random variables, we parameterize the likelihood as a mixture distribution

$$P(Y) = \pi_0 1_{Y \notin \mathcal{A}} P_c(Y) + \sum_{j=1}^{K} \pi_j 1_{Y = a_j}$$

where $\mathcal{A} = \{a_1, ..., a_K\}$ is the set of (discrete) atoms, π_j for $j = 0, ..., K$ forms a convex sum, and P_c is the density function of the continuous component. We have dropped the conditioning to simplify the notation.

Optimization For all the datasets, we use a 50/10/40 split for the training set, validation set, and test set. To preprocess the covariate (W) and the outcome (Y), we either standardize the data to have zero mean and unit variance, or normalize it so that the training data ranges from 0 to 1. We use the Adam optimizer to maximize the likelihood of the training data, and save the model with the best validation likelihood for evaluation and model selection. We perform grid search on the hyperparameters, and select the model with the best (early-stopped) validation likelihood and with a p-value passing 0.05 on the validation set.

Tunable Knobs After we fit a generative model to a dataset, we might like to get other models that are very similar but differ along important dimensions of interest. For example, this will allow us to test estimators in settings where there are positivity/overlap violations, where the causal effect is large/small, or where there is a lot of heterogeneity, no heterogeneity, etc. We currently have the following three knobs that we can turn to generate new but related distributions, after we've fit a model to a real dataset:

1. Positivity (how close $P(T = 1 \mid W)$ is to 0 or 1)
2. Size of causal effect

3. Degree of heterogeneity

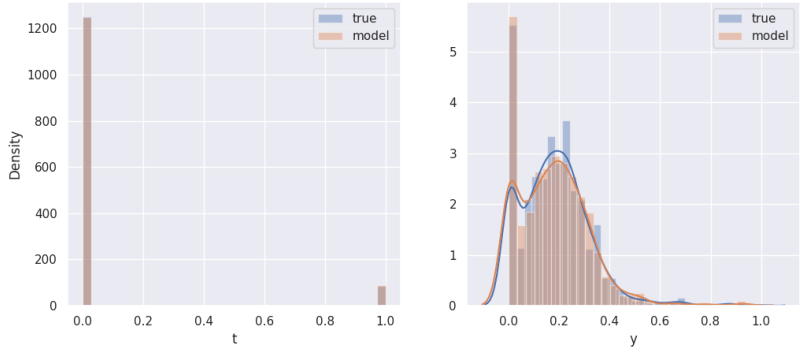
5. How Realistic is RealCause?

In this section, we show that RealCause produces realistic datasets that are very close to the real ones. For all datasets, we show that the distribution of our generative model $P_{\text{model}}(W, T, Y)$ is very close to the true distribution $P(W, T, Y)$. We show this by providing both visual comparisons and quantitative comparisons. We visually compare $P_{\text{model}}(T, Y)$ and $P(T, Y)$ using histograms and Gaussian kernel density estimation (see, e.g., Figure 1). We quantitatively compare $P_{\text{model}}(W, T, Y)$ and $P(W, T, Y)$ by running two-sample tests (Table 1).

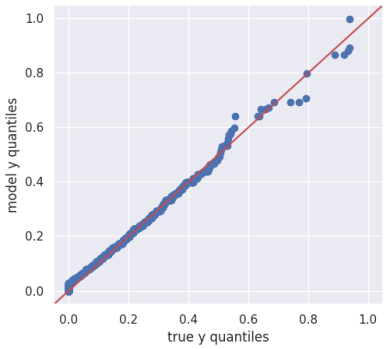
Two-sample tests evaluate the probability that a sample from $P_{\text{model}}(W, T, Y)$ and a sample from $P(W, T, Y)$ came from the same distribution, under the null hypothesis that $P_{\text{model}}(W, T, Y) \stackrel{d}{=} P(W, T, Y)$ (that the model distribution matches the true distribution). If that probability (p-value) is greater than some small value α such as 0.05, we say we have sufficient evidence to reject the null hypothesis that $P_{\text{model}}(W, T, Y) \stackrel{d}{=} P(W, T, Y)$ (i.e. the generative model is not as realistic as we would like it to be). This is how we operationalize the hypothesis that our modelled distributions are "realistic." Two-samples tests give us a way to falsify the hypothesis that our generative models are realistic.

However, two-sample tests do not work well in high-dimensions. Importantly, the power[3] of two-sample tests can decay with dimensionality (Ramdas et al., 2015) and W can have many dimensions in the datasets we consider. On the bright side, the treatment T and the outcome Y are each one-dimensional, so evaluating the statistical relationship between them is only a two-dimensional problem. This means that we might get more power from testing the hypothesis that $P_{\text{model}}(T, Y) \stackrel{d}{=} P(T, Y)$, even though this test will ignore W and its relationship to T and Y. Therefore, we run two-sample tests for both $P(T, Y)$ and $P(W, T, Y)$ (and the marginals). Tests that use $P(W, T, Y)$ could have more power because they use information about $P(T, Y \mid W)$

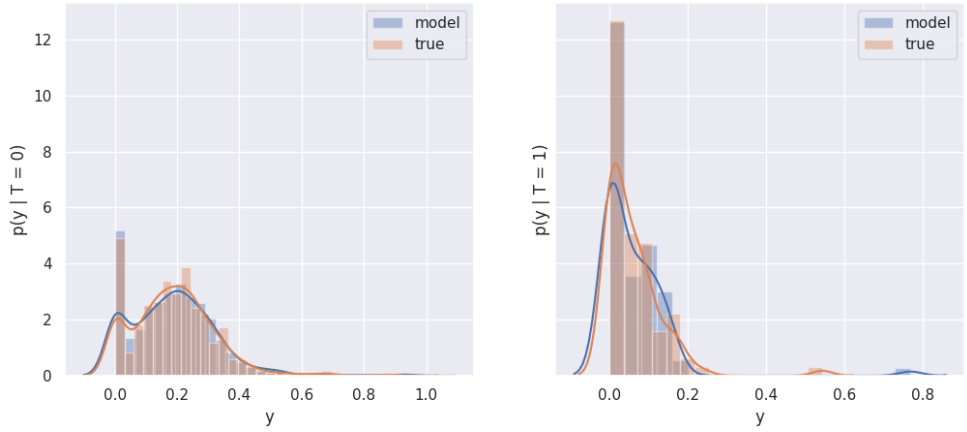
[3]For a fixed value of α, *power* is the probability of rejecting the null hypothesis, given that the null hypothesis is false.



(a) Marginal distributions $P(T)$ and $P_{\text{model}}(T)$ on the left and marginal distributions $P(Y)$ and $P_{\text{model}}(Y)$ on the right.



(b) Q-Q plot of $P_{\text{model}}(Y)$ and $P(Y)$.



(c) Histogram and kernel density estimate visualization of $P(Y \mid T)$ and $P_{\text{model}}(Y \mid T)$. Both graphs share the same y-axis. This graph, in combination with $P(T)$ and $P_{\text{model}}(T)$ in Figure 1a, gives complete visual information about $P(T, Y)$ and $P_{\text{model}}(T, Y)$.

Figure 1: Visualizations of how well the generative model models the real LaLonde PSID data.

(recall that $P(W) \stackrel{d}{=} P_{\text{model}}(W)$, by construction), whereas tests that use $P(T, Y)$ could have more power because they operate on a much lower dimensional space.

Datasets We fit eight datasets in total. We fit generative models to three real datasets: LaLonde PSID, LaLonde CPS (LaLonde, 1986) (we use Dehejia & Wahba (1999)'s version), and Twins[4] (Louizos et al., 2017). We additionally fit generative models to five popular semi-synthetic datasets: IHDP (Hill, 2011) and four LBIDD datasets (Shimoni et al., 2018). On all of these datasets, we can fit generative models to model the observational distribution. Then, with the semi-synthetic datasets, we can also check that our generative models give roughly the same ground-truth causal effects as existing popular synthetic benchmarks.

Visualization of Modeled LaLonde PSID Consider the LaLonde PSID dataset as our first example. Note that this is the dataset that Athey et al. (2019) had trouble fitting with a WGAN. We visualize $P_{\text{model}}(T)$ vs. $P(T)$ and $P_{\text{model}}(Y)$ vs. $P(Y)$ in Figures 1a and 1b. $P_{\text{model}}(W)$ and $P(W)$ are known to be the same distributions, by construction. We visualize $P_{\text{model}}(T, Y)$ vs. $P(T, Y)$ in Figure 1c. We provide similar visualizations of the other real datasets and corresponding similar models in Appendix A.

[4]The treatment selection mechanism for the Twins dataset is simulated. This is to ensure that there is some confounding, as the regular dataset might be unconfounded.

Univariate Statistical Tests The Kolmogorov-Smironov (KS) test is the most popular way to test the hypothesis that two samples come from the same distribution. The Epps-Singleton (ES) test is more well-suited for discrete distributions and can have higher power than the KS test (Epps & Singleton, 1986). We use the implementations of the KS and ES tests from *SciPy* (Virtanen et al., 2020). For all datasets, we report the p-values of the KS and ES tests for comparing the marginal distributions $P_{\text{model}}(Y)$ and $P(Y)$ and for comparing the marginal distributions $P_{\text{model}}(T)$ and $P(T)$ in the first section of Table 1. In all tests, the p-values are much larger than any reasonable value of α, so we fail to reject the null hypothesis that the generated data and the true data come from the same distribution. This means that our generative models are reasonably realistic, at least if we only look at the marginals.

Multivariate Statistical Test Extending the KS test to multiple dimensions is difficult. However, there are several multivariate tests such as the Friedman-Rafsky test (Friedman & Rafsky, 1979), k-nearest neighbor (kNN) test (Friedman & Rafsky, 1983), and energy test (Székely & Rizzo, 2013). We use the implementations of these tests in the *torch-two-sample* Python library (Djolonga, 2017). These are just permutation tests and can be conducted with any statistic, so we additionally run permutation tests with the Wasserstein-1 and Wasserstein-2 distance metrics. We run each test with 1000 permutations. We display the cor-

Table 1: Table of p-values for the various statistical hypothesis tests we run to test the null hypothesis that real data samples and samples from the generative model come from the same distribution. Large values (e.g. > 0.05) mean that we don't have statistically significant evidence that the real and generated data come different distributions, so we want to see large values. The first section is univariate tests, where KS stands for Kolmogorov-Smirnov and ES stands for Epps–Singleton. The second section is 2-dimensional tests to capture the dependence of Y on T, where FR stands for Friedman-Rafsky. The third section can be much higher dimensional tests whose power may suffer from the high dimensionality, but these tests may be able to pick up on the dependence of T and Y on W that the 2-dimensional tests cannot pick up on.

Test	LaLonde PSID	LaLonde CPS	Twins	IHDP	LBIDD Quad	LBIDD Exp
T KS	0.9995	1.0	0.9837	0.9290	0.5935	0.9772
T ES	0.6971	0.3325	0.7576	0.5587	0.8772	0.6975
Y KS	0.4968	1.0	0.8914	0.3058	0.2204	0.9146
Y ES	0.3069	0.1516	0.4466	0.3565	0.2264	0.7223
(T, Y) Wass1	0.6914	0.435	0.5088	0.2894	0.3617	0.4391
(T, Y) Wass2	0.6638	0.4356	0.4960	0.3365	0.4353	0.4709
(T, Y) FR	0.0	0.4004	0.5549	0.4761	0.8610	0.5773
(T, Y) kNN	0.0	0.4120	0.4318	0.5978	0.3166	0.3735
(T, Y) Energy	0.6311	0.4396	0.5053	0.3186	0.2371	0.4453
(W, T, Y) Wass1	0.4210	0.3854	0.4782	1.0	0.5191	0.4219
(W, T, Y) Wass2	0.5347	0.3660	0.4728	1.0	0.5182	0.4160
(W, T, Y) FR	0.2569	0.4033	0.5068	1.0	0.4829	0.4989
(W, T, Y) kNN	0.2270	0.4343	0.4919	1.0	0.5104	0.5101
(W, T, Y) Energy	0.5671	0.4177	0.5263	0.9409	0.5104	0.4423
$\lvert W \rvert$ (n covariates)	8	8	75	25	177	177

responding p-values in the last two sections of Table 1. For all tests except the FR and kNN (T, Y) test on the LaLonde PSID dataset,[5] the p-values are much larger than any reasonable value of α. However, we might be worried that these multivariate two-sample tests don't have enough power when we include the higher-dimensional W.

Demonstration of Statistical Power via Linear Baselines We demonstrate that these test do have a decent amount of statistical power (probability of rejecting the null when P_{model} and P differ) by fitting a linear Gaussian model to the data and displaying the corresponding p-values in Table 2. Even when W is high-dimensional, we are still able to reject the linear models as realistic. For example, we clearly have p-values that are below most reasonable values of α for the LaLonde PSID, LBIDD Quad, and LBBIDD Exp datasets. As we might expect, for high-dimensional W such as in the LBIDD datasets, the (T, Y) tests have enough power to reject the null hypothesis because they operate in only two dimensions, whereas the (W, T, Y) tests do not because their power suffers from the high-dimensionality (179 dimensions). The LaLonde CPS data is an example where it can be useful to include W in the statistical test; all of the p-values for the (T, Y) tests are *above* $\alpha = .075$, whereas all but one of the p-values for the (W, T, Y) tests are *below* $\alpha = .075$. Our p-values for the Twins dataset are quite high, but this is not due to these tests not having enough power. Rather, it is because the Twins dataset is well modeled by a linear model: T and Y are both binary (two parameters) and W is 75-dimensional, so it makes sense that we can linearly predict these two parameters from 75 dimensions. You can see the plots for how well the linear model fit Twins in Figures 4c and 4d in Appendix A. Similarly, the p-values for IHDP are so high because a combination of the fact that the IHDP data is reasonably well fit by the linear model (see Figures 5d to 5f) and the fact that the IHDP tests can't have that much power since the IHDP dataset is much smaller than the other datasets.

[5]The p-values of exactly zero for the (T, Y) LaLonde PSID tests indicate that something fishy is going on; we are looking into this.

Realistic Causal Effects We also show that our generative model admits causal effect estimates that roughly match those of the popular semi-synthetic benchmarks IHDP and LBIDD. For each of these datasets, we report the true ATE, our generative model's ATE estimate, the corresponding absolute bias, and a measure that takes into account causal effect heterogeneity: PEHE (see Equation (20)). We report these values in Table 3. The values in the table indicate that our model accurately models the causal effects. The one number that is relatively high relative to the others is

Table 2: Table of p-values for the various statistical hypothesis tests we run to test the null hypothesis that real data samples and samples from a *linear* Gaussian generative model come from the same distribution. Small values (e.g. < 0.05) mean that these tests have enough power to detect that the real data comes from a different distribution than the distribution generated by our linear Gaussian generative model.

Test	LaLonde PSID	LaLonde CPS	Twins	IHDP	LBIDD Quad	LBIDD Exp
(T, Y) Wass1	0.0304	0.1500	0.5004	0.2019	0.2009	0.0456
(T, Y) Wass2	0.0123	0.0797	0.4924	0.1636	0.4277	0.1314
(T, Y) FR	0.0	0.0776	0.5581	0.2825	0.0	0.0014
(T, Y) kNN	0.0	0.1808	0.4541	0.4183	0.0	0.0023
(T, Y) Energy	0.0482	0.1620	0.5094	0.2249	0.0002	0.0551
(W, T, Y) Wass1	0.0470	0.0671	1.0	1.0	0.4917	0.5245
(W, T, Y) Wass2	0.4001	0.0624	0.9966	1.0	0.4782	0.5204
(W, T, Y) FR	0.1333	0.0525	0.9992	1.0	0.7655	0.6979
(W, T, Y) kNN	0.5136	0.0711	1.0	1.0	0.8953	0.8416
(W, T, Y) Energy	0.1080	0.2863	0.7389	0.8935	0.5099	0.5142
$\lvert W \rvert$ (n covariates)	8	8	75	25	177	177

Table 3: True causal effects, corresponding estimates from our generative model, and associated error.

	IHDP	LBIDD Quad	LBIDD Exp	LBIDD Log	LBIDD Linear
True ATE	4.0161	2.5437	-0.6613	0.0549	1.8592
ATE estimate	4.1908	2.4910	-0.6608	0.0555	1.7177
ATE abs bias	0.1747	0.0527	0.0004	0.0005	0.1415
PEHE	51.5279	0.1554	0.0225	0.0151	0.1367

the PEHE for IHDP; this is because the training sample for IHDP is only 374 examples. Note that we cannot do this for the real datasets because the true causal effects of the real datasets are unknown, due to the fundamental problem of causal inference.

6. Causal Estimators

6.1. Nonparametric Estimators

6.1.1. Conditional outcome modeling

We rewrite Equation 4 using the mean conditional outcome $\mu(t, w)$ from Equation 8:

$$\tau = \mathbb{E}_W \left[\mathbb{E}[Y|T=1, W] - \mathbb{E}[Y|T=0, W]\right] \tag{10}$$
$$= \mathbb{E}_W \left[\mu(1, W) - \mu(0, W)\right] \tag{11}$$

By plugging in an arbitrary model $\hat{\mu}$, we get the following:

$$\hat{\tau} = \frac{1}{m} \sum_w \left(\hat{\mu}(1, w) - \hat{\mu}(0, w)\right) , \tag{12}$$

where m is the number of observations. This is a *nonparametric estimator* of τ, since we have not specified any parametric assumptions for $\hat{\mu}$.[6] We call the class of estimators that result from using an arbitrary model for $\hat{\mu}$ *conditional outcome model* (COM) estimators because they model the conditional outcome μ and adjust for the confounders W. These estimators also go by other names such as G-computation estimators (see, e.g., Snowden et al., 2011) standardization (Hernán & Robins, 2020) and S-learner (Künzel et al., 2019). The models $\hat{\mu}(t, w)$ also provide estimates of IATEs by simply plugging in to Equation 8:

$$\hat{\tau}(w) = \hat{\mu}(1, w) - \hat{\mu}(0, w) . \tag{13}$$

6.1.2. Grouped conditional outcome modeling

Rather than modeling μ with a single model $\hat{\mu}$, we could model μ with a model for each value of treatment. Specifically, we can model $\mu(1, w)$ with a model $\hat{\mu_1}(w)$, and we can model $\mu(0, w)$ with another model $\hat{\mu_0}(w)$. Then, we can estimate ATEs and IATEs as follows:

$$\hat{\tau} = \frac{1}{m} \sum_w \left(\hat{\mu_1}(w) - \hat{\mu_0}(w)\right) \tag{14}$$
$$\hat{\tau}(w) = \hat{\mu_1}(w) - \hat{\mu_0}(w) . \tag{15}$$

We call the class of estimators that model μ via $\hat{\mu_1}(w)$ and $\hat{\mu_0}(w)$ *grouped conditional outcome model* (GCOM) estimators. This is also sometimes called the T-learner (Künzel et al., 2019).

[6] Künzel et al. (2019) refer to nonparametric estimators as "meta-learners."

6.1.3. X-learner

The X-learner (Künzel et al., 2019) builds on stratified conditional outcome adjustment. After estimating $\hat{\mu_0}$ and $\hat{\mu_1}$, the X-learner does not immediately use these to estimate causal effects. Rather, it uses $\hat{\mu_0}$ with the potential outcomes $Y(1)$ that are observed in the treatment group and uses $\hat{\mu_1}$ with the potential outcomes $Y(0)$ that are observed in the control group:

$$\widetilde{\tau}_i^1 \triangleq Y(1) - \hat{\mu_0}(W_i)$$
$$\widetilde{\tau}_i^0 \triangleq \hat{\mu_1}(W_i) - Y(0)$$

This crossing between the treatment group and control group ($Y(1)$ vs. $\hat{\mu_0}(W_i)$ and $Y(0)$ vs. $\hat{\mu_1}(W_i)$) is why it is called the X-learner. It then regresses $\widetilde{\tau}_i^1$ on W in the treatment group to get $\hat{\tau}^1(w)$. Similarly, it regresses $\widetilde{\tau}_i^0$ on W in the control group to get $\hat{\tau}^0(w)$. The final estimator is then a convex combination of these two estimators:

$$\hat{\tau}(w) = \alpha(w)\hat{\tau}^0(w) + (1 - \alpha(w))\hat{\tau}^1(w) \tag{16}$$

In this paper, we use a model for the propensity score for $\alpha(w)$ (i.e. we use $\alpha(w) = \hat{e}(w)$). Künzel et al. (2019) observe that is a good choice for $\alpha(w)$.

6.1.4. Inverse probability weighting

We can also adjust for confounding by reweighting the population such that each observation is weighted by $\frac{1}{P(t_i|w_i)}$. This reweighted population is referred to as the *pseudo-population* (Hernán & Robins, 2020). This general technique (Horvitz & Thompson, 1952) is known as *inverse probability weighting* (IPW). It can be shown that

$$\tau = \mathbb{E}\left[\frac{I(T=1)Y}{P(T|W)} - \frac{I(T=0)Y}{P(T|W)}\right] , \tag{17}$$

where I denotes an indicator random variable, which takes the value 1 if its argument is true and takes the value 0 otherwise. Now, a natural estimator is a plug-in estimator for this estimand. Note that $P(T = 1|W = w)$ is the propensity score $e(w)$. We can then estimate τ using an arbitrary model $\hat{e}(w)$ via the following plug-in estimator:

$$\hat{\tau} = \frac{1}{m} \sum_w \left(\frac{I(T=1)Y}{\hat{e}(w)} - \frac{I(T=0)Y}{1 - \hat{e}(w)}\right) \tag{18}$$

The weights $\frac{1}{\hat{e}(w)}$ for the treated units and $\frac{1}{1-\hat{e}(w)}$ for the untreated units can sometimes be very large, leading to high variance. Therefore, it is common to trim them. We consider estimators that trim them when $\hat{e}(w) < 0.01$ or when $\hat{e}(w) > 0.99$.

Another option is to "stabilize" the weight by multiplying them by fractions that are independent of W and sum to 1.

A natural option is $P(T)$. This yields the *stabilized IPW estimator*:

$$\hat{\tau} = \frac{1}{m} \sum_w \left(\frac{I(T=1)P_T(1)Y}{\hat{e}(w)} - \frac{I(T=0)P_T(0)Y}{1-\hat{e}(w)} \right) \tag{19}$$

6.2. Outcome and Propensity Score Models

We model $\hat{\mu}(t,w)$, $\hat{\mu_1}(w)$, $\hat{\mu_0}(w)$, and $\hat{e}(w)$ using a large variety of models available in *scikit-learn* (Pedregosa et al., 2011).

We consider the following models for $\hat{\mu}$, $\hat{\mu_1}$, and $\hat{\mu_0}$: ordinary least-squares (OLS) linear regression, lasso regression, ridge regression, elastic net, support-vector machines (SVMs) with radial basis function (RBF) kernels, SVMs with sigmoid kernels, linear SVMs, SVMs with standardized inputs (because SVMs are sensitive to input scale), kernel ridge regression, and decision trees.

We consider the following models for $\hat{e}$: vanilla logistic regression, logistic regression with L2 regularization, logistic regression with L1 regularization, logistic regression with variants on the optimizer such as liblinear or SAGA, k-nearest neighbors (kNN), decision trees, Gaussian Naive Bayes, and quadratic discriminant analysis (QDA).

7. Benchmarking Causal Estimators

We get 66 estimators by taking the Cartesian product of the nonparametric estimators described in Section 6.1 and the models described in Section 6.2. We use *causallib* (Shimoni et al., 2019) for the nonparametric estimators and *scikit-learn* (Pedregosa et al., 2011) for the models. We benchmark these estimators against the data from our LaLonde PSID generative model.

For now, we choose the hyperparameters by trying out 10 different values for the most important hyperparameter for a given model and choosing the one that corresponds to the best predictive accuracy (does not necessarily correspond to the best accuracy for causal effects). For many hyperparameters, this search is done in log-space.

7.1. Evaluation Metrics

We produce 100 different samplings of the modeled dataset. We denote the i^{th} dataset by D_i. These are what we average over to approximate the bias, standard deviation, and root mean squared error (RMSE) of the ATE estimates $\hat{\tau}$. For example, we approximate the bias $\mathbb{E}[\hat{\tau}] - \tau$ with a Monte Carlo average over the 100 data samplings.

To assess the performance of estimators for predicting IATEs, we use the PEHE (precision in estimation of heterogeneous effects) (Hill, 2011). For a given dataset D_i, the PEHE is defined as follows:

$$\text{PEHE} \triangleq \sqrt{\frac{1}{m} \sum_w \left(\hat{\tau}(w) - \tau(w)\right)^2} \tag{20}$$

We report the mean PEHE across all data samplings. The IPW estimators do not estimate IATEs, so we do not report the PEHE for them.

7.2. Benchmark Results

7.2.1. Conditional outcome adjustment

We report the results of the conditional outcome adjustment estimators (Section 6.1.1) in Table 4. In all columns except the Bias($\hat{\tau}$) column, lower values are better. To give an idea for scale, the true ATE is -7763.77.

Conditional outcome adjustment with linear regression for the outcome model (first row in Table 4) is equivalent to just fitting a linear regression and taking the coefficient on T for the ATE estimate. We can see that this performs worse than any of the non-OLS-regression estimators. Because this method actually assumes a homogeneous treatment effect ($\hat{\tau}(w)$ is the same for all w), any reasonable estimator should get a lower value for mean PEHE than linear regression gets: 16139.

SVMs without standardized inputs all do poorly because they ignore T because it is on a much smaller scale than W. This causes them to yield ATE estimates of zero always. For some reason, decision trees also ignore T and yield ATE estimates of zero as well.

Clearly, SVMs with RBF kernels and standardized input perform the best of the outcome models we tried, when using conditonal outcome adjustment. They have the smallest absolute bias, RMSE, and mean PEHE.

7.2.2. Stratified conditional outcome adjustment

In Table 5, we show results for stratified conditional outcome adjustment (Section 6.1.2). For now, we use the same outcome model to model both $\mu_0(w)$ and $\mu_1(w)$, but you could imagine that we can take the Cartesian product of the outcome models.

The first important thing to notice when we switch to stratified conditional outcome adjustment is that many of the linear models that performed quite poorly when they had to model the interaction of T and W now perform quite well. This means that while $\mu(t,w)$ must be highly nonlinear, $\mu_0(w)$ and $\mu_1(w)$ are noticeably more well approximated by linear functions. In fact, OLS linear regression even ended up with the lowest absolute bias of all the outcome models.

Table 4: Performance of conditional outcome adjustment with several different outcome models. Lower values are better in all columns except the Bias($\hat{\tau}$) column.

Outcome Model	Bias($\hat{\tau}$)	|Bias($\hat{\tau}$)|	Std($\hat{\tau}$)	RMSE($\hat{\tau}$)	Mean PEHE
LinearRegression	8719.02	8719.02	926.51	8768.11	16139.45
LinearRegression_interact	-1105.91	1105.91	4640.40	4770.37	15328.92
LinearRegression_degree2	-1232.79	1232.79	7834.64	7931.04	17433.21
LinearRegression_degree3	-10960.65	10960.65	53531.19	54641.78	127062.12
Lasso	7791.13	7791.13	112.17	7791.93	15625.94
Ridge	7815.57	7815.57	110.54	7816.35	15638.48
ElasticNet	7799.52	7799.52	85.56	7799.99	15630.13
SVM_rbf	7763.77	7763.77	0.00	7763.77	15611.64
SVM_sigmoid	7763.77	7763.77	0.00	7763.77	15611.64
LinearSVM	7763.77	7763.77	0.00	7763.77	15611.64
Standardized_SVM_rbf	1501.73	1501.73	1467.44	2099.66	12847.54
Standardized_SVM_sigmoid	3912.96	3912.96	622.98	3962.24	13748.54
Standardized_LinearSVM	7877.80	7877.80	472.03	7891.93	15675.47
KernelRidge	7883.02	7883.02	109.48	7883.78	15673.24
DecisionTree	7763.77	7763.77	0.00	7763.77	15611.64

However, linear SVMs with standardized input outperformed linear regression (using RMSE and mean PEHE as our performance metrics). This is because, while the SVM had much higher bias than linear regression, it has much lower variance. Still, it turns out a linear model (linear SVM) performed the best when using stratified conditional outcome adjustment. However, the RBF SVM using regular conditional outcome adjustment is still the best of the estimators we have examined so far.

7.2.3. Inverse probability weighting

In Table 6 in Appendix B, we present results for IPW estimators (Section 6.1.4) using a variety of different propensity score models. We also consider estimators that trim (throw out) observations with propensity scores that are less than 0.01 and greater than 0.99. We additionally, consider estimators that use weight stabilization.

The main thing to notice is that weight trimming helps substantially. All of the estimators that achieve RMSE's below 4000 use weight trimming. Weight trimming even takes quadratic discriminant analysis (QDA) from giving the worst estimates (RMSE of 25075.21) to giving the best estimates (RMSE of 3544.91).

Potentially unsurprisingly, we see that weight stabilization has absolutely no effect. All estimators achieve exactly the same RMSE with weight stabilizatin as they do without weight stabilization.

8. Discussion and Extensions

The main focus of this working paper is to show that our generative models are realistic. We've only evaluated 66 estimators on a single dataset, but we plan to have a much more comprehensive analysis in the near future. We plan to add many more estimators, evaluate across all the datasets with several different settings of the knobs each, use more advanced hyperparameter tuning, and generally produce a more complete comparison of the estimators in the near future. Additionally, we plan to add a larger variety of generative models to assess the stability of rankings to the choice of generative model.

Conditional outcome modeling with an RBF SVM for the outcome model (using standardized inputs) performed the best. However, it very well may not perform the best when we include more nonparametric estimators and more flexible models such as random forests, neural networks, and boosting. Additionally, as we add more datasets, we would be surprised if a single estimator performed the best over all datasets. Ideally, we will be able to predict which estimators will perform best for causal estimands, using purely statistical metrics.

Table 5: Performance of *stratified* conditional outcome adjustment with several different outcome models. The same type of model is used for modeling both $\mu_1(w)$ and $\mu_0(w)$. Lower values are better in all columns except the Bias($\hat{\tau}$) column.

Outcome Model	Bias($\hat{\tau}$)	|Bias($\hat{\tau}$)|	Std($\hat{\tau}$)	RMSE($\hat{\tau}$)	Mean PEHE
LinearRegression	206.09	206.09	4173.18	4178.27	14029.37
LinearRegression_interact	1969.19	1969.19	52960.32	52996.92	103071.79
LinearRegression_degree2	-12629.66	12629.66	99567.70	100365.51	152041.13
LinearRegression_degree3	-277998100.27	277998100.27	2685023196.87	2699376318.98	793876117.95
Lasso	-328.59	328.59	4124.87	4137.94	13983.15
Ridge	-1289.03	1289.03	4009.80	4211.90	13985.43
ElasticNet	-651.52	651.52	4068.18	4120.02	13910.22
SVM_rbf	-4985.77	4985.77	1665.74	5256.67	14942.79
SVM_sigmoid	-7820.26	7820.26	478.31	7834.87	15039.87
LinearSVM	3538.38	3538.38	3394.66	4903.45	13789.14
Standardized_SVM_rbf	-5285.18	5285.18	1022.99	5383.28	14969.84
Standardized_SVM_sigmoid	-6578.45	6578.45	660.55	6611.53	14485.85
Standardized_LinearSVM	-2496.47	2496.47	2171.54	3308.77	13546.05
KernelRidge	-904.50	904.50	3841.08	3946.14	13911.78
DecisionTree	1316.59	1316.59	5723.28	5872.76	18605.27

References

Abadie, Alberto & Guido W. Imbens (2011). "Bias-Corrected Matching Estimators for Average Treatment Effects". In: *Journal of Business & Economic Statistics*.

Arjovsky, Martin, Soumith Chintala & Léon Bottou (June 2017). "Wasserstein Generative Adversarial Networks". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup & Yee Whye Teh. Proceedings of Machine Learning Research. International Convention Centre, Sydney, Australia: PMLR.

Athey, Susan et al. (2019). *Using Wasserstein Generative Adversarial Networks for the Design of Monte Carlo Simulations*.

Dehejia, Rajeev H. & Sadek Wahba (1999). "Causal Effects in Nonexperimental Studies: Reevaluating the Evaluation of Training Programs". In: *Journal of the American Statistical Association*.

Djolonga, Josip (2017). *A PyTorch library for differentiable two-sample tests*. https://github.com/josipd/torch-two-sample.

Dorie, Vincent et al. (Feb. 2019). "Automated versus Do-It-Yourself Methods for Causal Inference: Lessons Learned from a Data Analysis Competition". In: *Statist. Sci.*

Epps, T.W. & Kenneth J. Singleton (1986). "An omnibus test for the two-sample problem using the empirical characteristic function". In: *Journal of Statistical Computation and Simulation*.

Franklin, Jessica et al. (Apr. 2014). "Plasmode Simulation for the Evaluation of Pharmacoepidemiologic Methods in Complex Healthcare Databases". In: *Computational Statistics & Data Analysis*.

Friedman, Jerome H. & Lawrence C. Rafsky (July 1979). "Multivariate Generalizations of the Wald-Wolfowitz and Smirnov Two-Sample Tests". In: *Ann. Statist.*

— (June 1983). "Graph-Theoretic Measures of Multivariate Association and Prediction". In: *Ann. Statist.*

Goodfellow, Ian et al. (2014). "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani et al. Curran Associates, Inc.

Hahn, P. Richard, Vincent Dorie & Jared S. Murray (2019). *Atlantic Causal Inference Conference (ACIC) Data Analysis Challenge 2017*.

Hernán, Miguel A & James M Robins (2020). *Causal Inference: What If*. Boca Raton: Chapman & Hall/CRC.

Hill, Jennifer L. (2011). "Bayesian Nonparametric Modeling for Causal Inference". In: *Journal of Computational and Graphical Statistics*.

Hill, Jennifer L., Jerome P. Reiter & Elaine L. Zanutto (2004). "A Comparison of Experimental and Observational Data Analyses". In: *Applied Bayesian Modeling and Causal Inference from Incomplete-Data Perspectives*. John Wiley & Sons, Ltd. Chap. 5.

Holland, Paul W. (1986). "Statistics and Causal Inference". In: *Journal of the American Statistical Association*.

Horvitz, D. G. & D. J. Thompson (1952). "A Generalization of Sampling Without Replacement from a Finite Universe". In: *Journal of the American Statistical Association*.

Huang, Chin-Wei et al. (2018). "Neural Autoregressive Flows". In: *International Conference on Machine Learning*.

Huber, Martin, Michael Lechner & Conny Wunsch (2013). "The performance of estimators based on the propensity score". In: *Journal of Econometrics*.

Imbens, Guido W. & Donald B. Rubin (2015). *Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction*. Cambridge University Press.

Kallus, Nathan, Aahlad Manas Puli & Uri Shalit (2018). "Removing Hidden Confounding by Experimental Grounding". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Curran Associates, Inc.

Knaus, Michael C., Michael Lechner & Anthony Strittmatter (2018). *Machine Learning Estimation of Heterogeneous Causal Effects: Empirical Monte Carlo Evidence*.

Künzel, Sören R. et al. (2019). "Metalearners for estimating heterogeneous treatment effects using machine learning". In: *Proceedings of the National Academy of Sciences*.

LaLonde, Robert J. (1986). "Evaluating the Econometric Evaluations of Training Programs with Experimental Data". In: *The American Economic Review*.

Lechner, Michael & Conny Wunsch (2013). "Sensitivity of matching-based program evaluations to the availability of control variables". In: *Labour Economics*.

Louizos, Christos et al. (2017). "Causal Effect Inference with Deep Latent-Variable Models". In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon et al. Curran Associates, Inc.

Morgan, Stephen L. & Christopher Winship (2014). *Counterfactuals and Causal Inference: Methods and Principles for Social Research*. 2nd ed. Analytical Methods for Social Research. Cambridge University Press.

Neal, Brady (2020). *Introduction to Causal Inference*.

Pearl, Judea (1994). "A Probabilistic Calculus of Actions". In: *ArXiv*.

— (2009). *Causality*. Cambridge University Press.

— (2019). "On the Interpretation of do(x)". In: *Journal of Causal Inference*.

Pearl, Judea, Madelyn Glymour & Nicholas P Jewell (2016). *Causal inference in statistics: A primer*. John Wiley & Sons.

Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research*.

Ramdas, Aaditya et al. (2015). "On the Decreasing Power of Kernel and Distance Based Nonparametric Hypothesis Tests in High Dimensions". In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. AAAI'15. Austin, Texas: AAAI Press.

Robins, James (1986). "A new approach to causal inference in mortality studies with a sustained exposure period—application to control of the healthy worker survivor effect". In: *Mathematical Modelling*.

Rubin, Donald B. (1974). "Estimating causal effects of treatments in randomized and nonrandomized studies." In: *Journal of educational Psychology*.

Schuler, Alejandro et al. (2017). *Synth-Validation: Selecting the Best Causal Inference Method for a Given Dataset*.

Shadish, William R., M. H. Clark & Peter M. Steiner (2008). "Can Nonrandomized Experiments Yield Accurate Answers? A Randomized Experiment Comparing Random and Nonrandom Assignments". In: *Journal of the American Statistical Association*.

Shalit, Uri, Fredrik D Johansson & David Sontag (2017). "Estimating individual treatment effect: generalization bounds and algorithms". In: *International Conference on Machine Learning*. PMLR.

Shi, Claudia, David Blei & Victor Veitch (2019). "Adapting neural networks for the estimation of treatment effects". In: *Advances in Neural Information Processing Systems*.

Shimoni, Y. et al. (2018). "Benchmarking Framework for Performance-Evaluation of Causal Inference Analysis". In: *ArXiv preprint arXiv:1802.05046*.

Shimoni, Yishai et al. (2019). *An Evaluation Toolkit to Guide Model Selection and Cohort Definition in Causal Inference*.

Snowden, Jonathan M., Sherri Rose & Kathleen M. Mortimer (Mar. 2011). "Implementation of G-Computation on a Simulated Data Set: Demonstration of a Causal Inference Technique". In: *American Journal of Epidemiology*.

Spirtes, Peter, Clark Glymour & Richard Scheines (Jan. 1993). *Causation, Prediction, and Search*.

Székely, Gábor J. & Maria L. Rizzo (2013). "Energy statistics: A class of statistics based on distances". In: *Journal of Statistical Planning and Inference*.

Turner, Ryan & Brady Neal (2018). "How well does your sampler really work?" In: *Uncertainty in Artificial Intelligence*. AUAI Press.

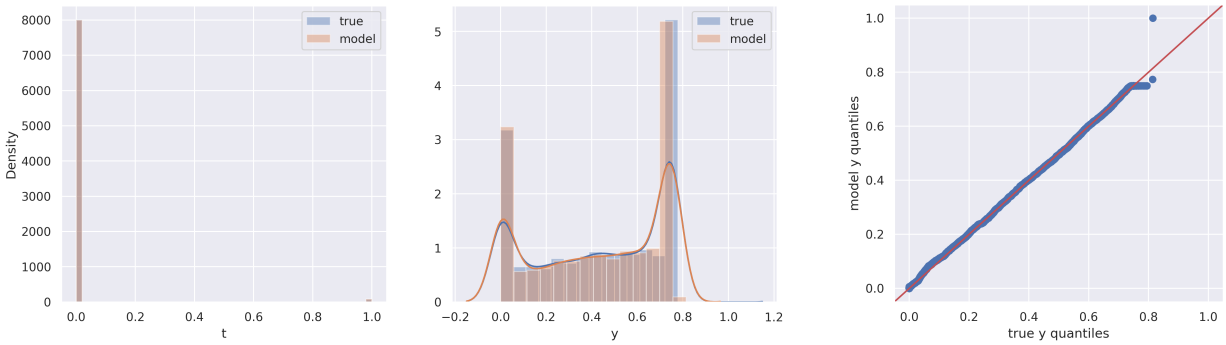
Virtanen, Pauli et al. (2020). "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods*.

Wendling, T. et al. (2018). "Comparing methods for estimation of heterogeneous treatment effects using observational data from health care databases". In: *Statistics in Medicine*.

Appendices

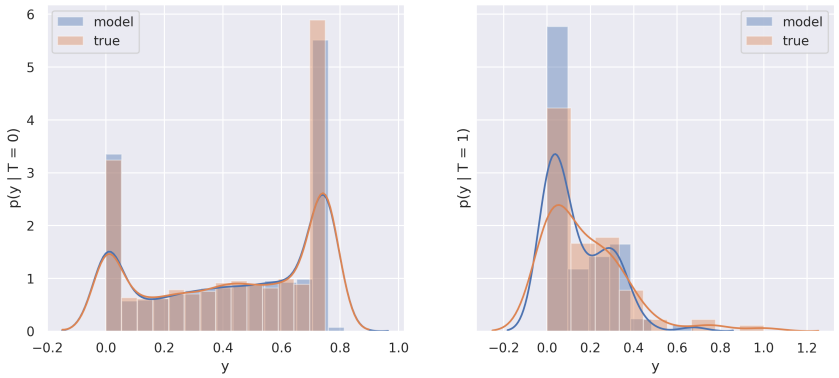
A. Visual Comparisons of Generated Distribution vs. Real Distributions

In this appendix, we provide the visualizations of the sigmoidal flows and the baseline linear generative models for each dataset. Each figure takes up a single page and corresponds to a single dataset. For each figure, the first half of the plots are for the sigmoidal flow and the second half of the plots are for the linear model. Because each figure takes up a whole page, the figures begin on the next page.

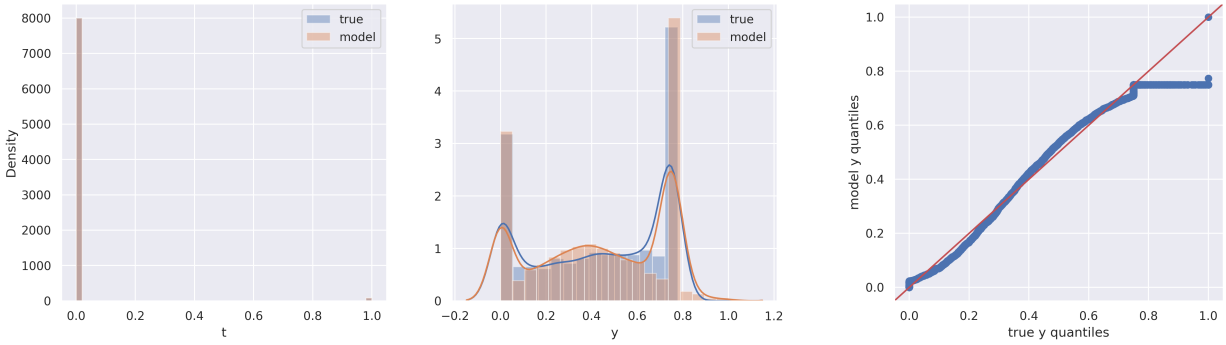


(a) Marginal distributions $P(T)$ and $P_{\text{model}}(T)$ on the left and marginal distributions $P(Y)$ and $P_{\text{model}}(Y)$ on the right.

(b) Q-Q plot of $P_{\text{model}}(Y)$ and $P(Y)$.

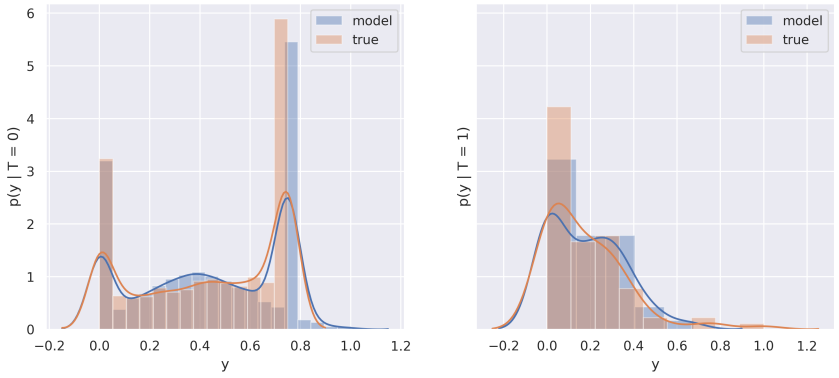


(c) Histogram and kernel density estimate visualization of $P(Y \mid T)$ and $P_{\text{model}}(Y \mid T)$. Both graphs share the same y-axis.



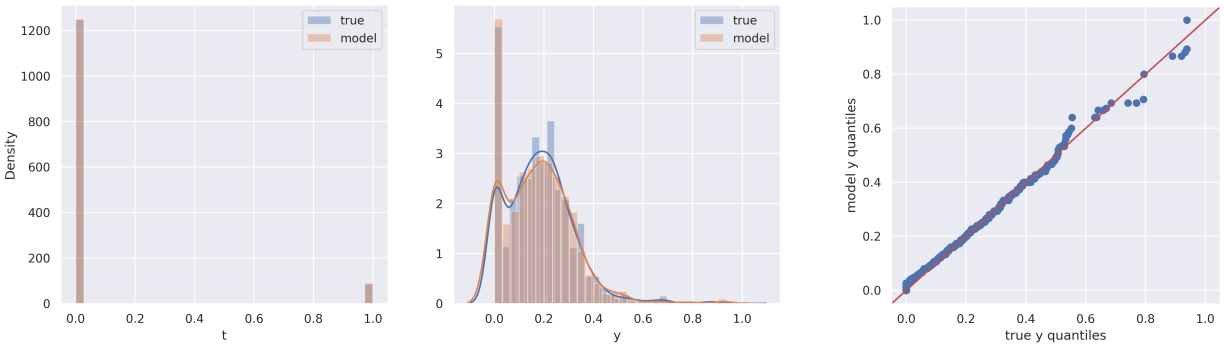
(d) Marginal distributions $P(T)$ and $P_{\text{model}}(T)$ on the left and marginal distributions $P(Y)$ and $P_{\text{model}}(Y)$ on the right.

(e) Q-Q plot of $P_{\text{model}}(Y)$ and $P(Y)$.



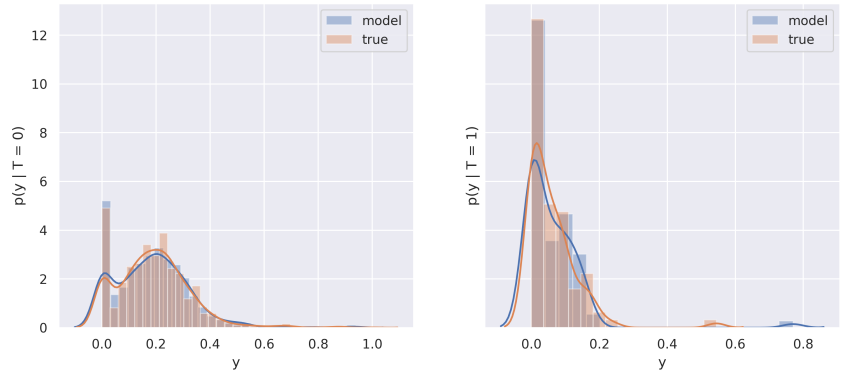
(f) Histogram and kernel density estimate visualization of $P(Y \mid T)$ and $P_{\text{model}}(Y \mid T)$. Both graphs share the same y-axis.

Figure 2: LaLonde CPS – Visualizations of how well the generative model models the real data. Figures (a) - (c) are visualizations of the sigmoidal flow model. Figures (d) - (f) are visualizations of the baseline linear Gaussian model.

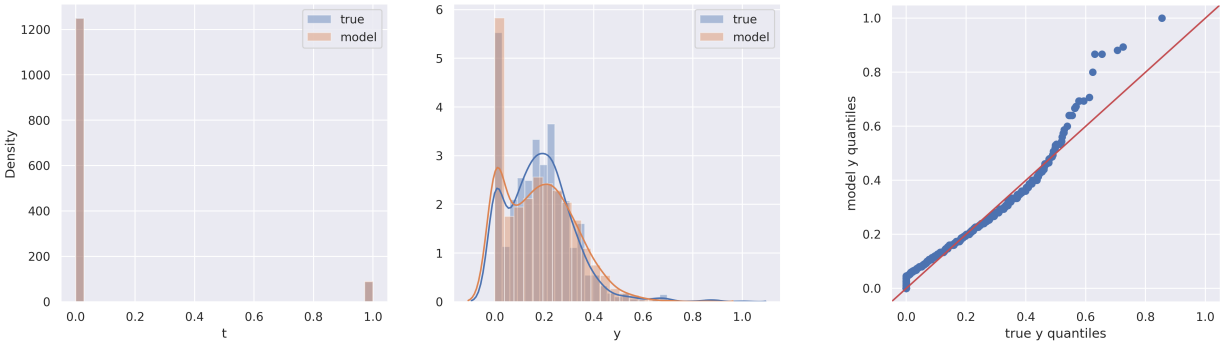


(a) Marginal distributions $P(T)$ and $P_{\text{model}}(T)$ on the left and marginal distributions $P(Y)$ and $P_{\text{model}}(Y)$ on the right.

(b) Q-Q plot of $P_{\text{model}}(Y)$ and $P(Y)$.

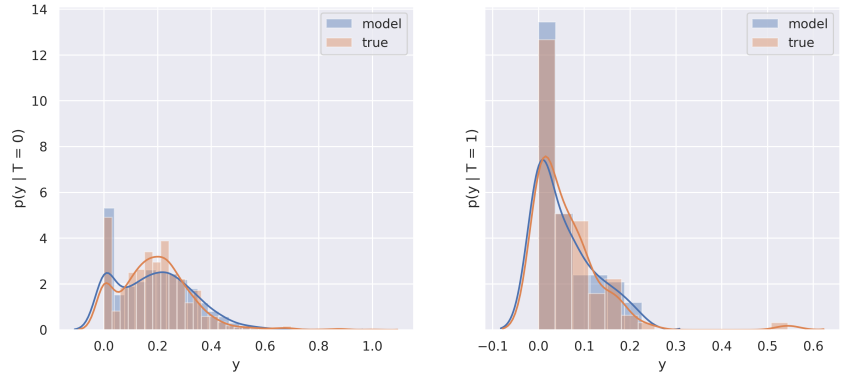


(c) Histogram and kernel density estimate visualization of $P(Y \mid T)$ and $P_{\text{model}}(Y \mid T)$. Both graphs share the same y-axis.



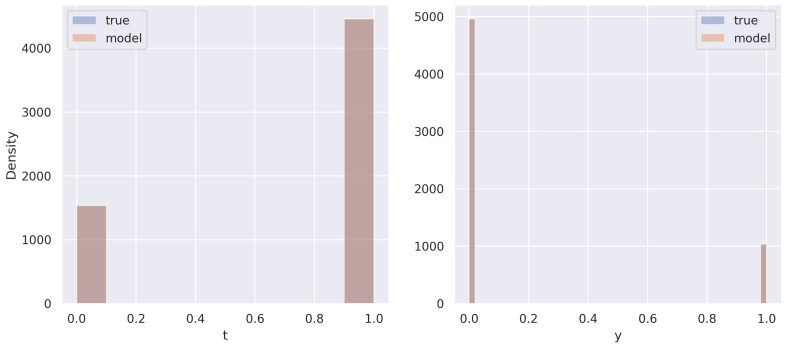
(d) Marginal distributions $P(T)$ and $P_{\text{model}}(T)$ on the left and marginal distributions $P(Y)$ and $P_{\text{model}}(Y)$ on the right.

(e) Q-Q plot of $P_{\text{model}}(Y)$ and $P(Y)$.

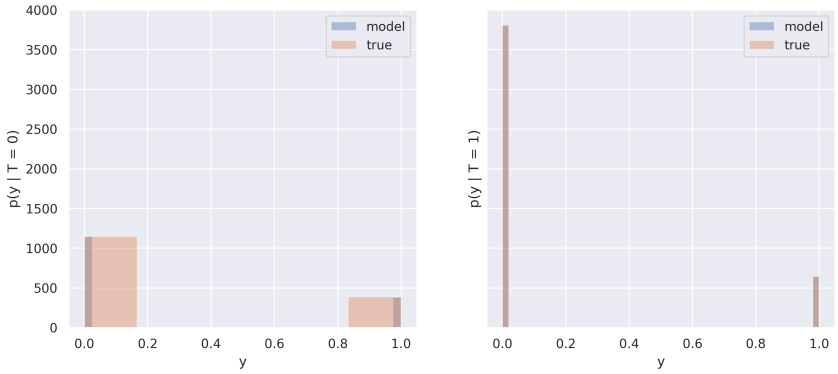


(f) Histogram and kernel density estimate visualization of $P(Y \mid T)$ and $P_{\text{model}}(Y \mid T)$. Both graphs share the same y-axis.

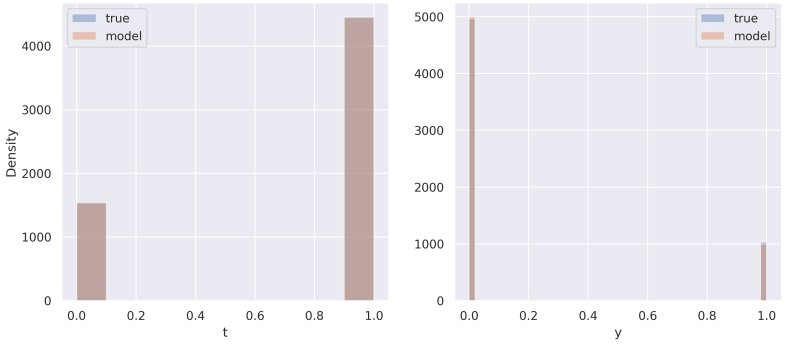
Figure 3: LaLonde PSID – Visualizations of how well the generative model models the real data. Figures (a) - (c) are visualizations of the sigmoidal flow model. Figures (d) - (f) are visualizations of the baseline linear Gaussian model.



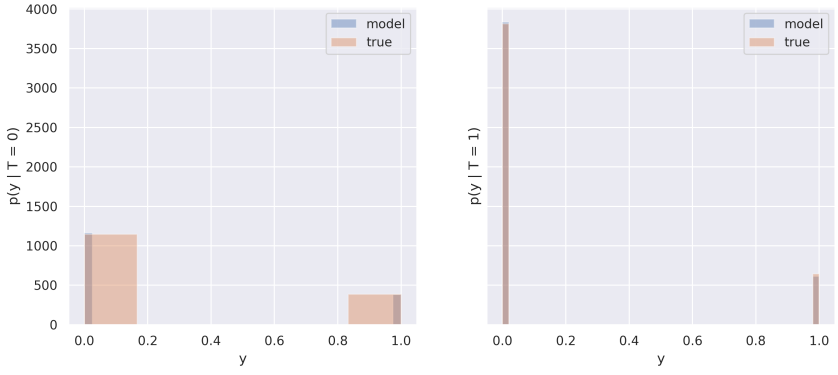
(a) Marginal distributions $P(T)$ and $P_{\text{model}}(T)$ on the left and marginal distributions $P(Y)$ and $P_{\text{model}}(Y)$ on the right.



(b) Histogram and kernel density estimate visualization of $P(Y \mid T)$ and $P_{\text{model}}(Y \mid T)$. Both graphs share the same y-axis.

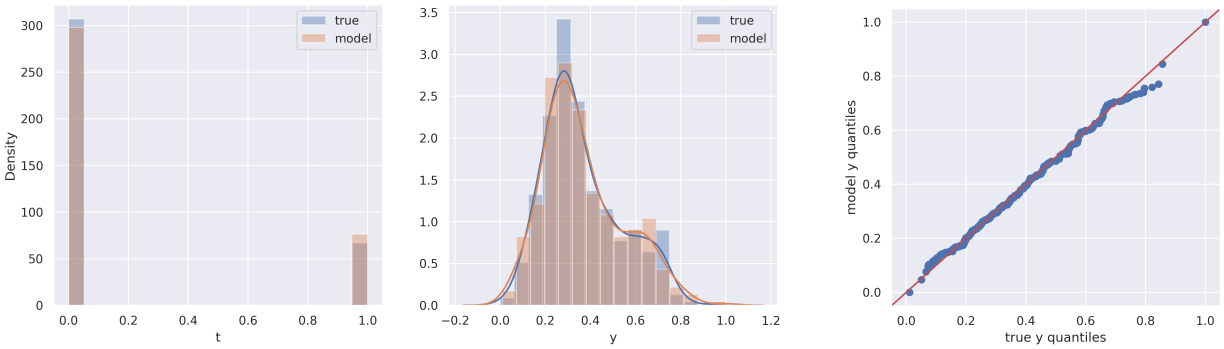


(c) Marginal distributions $P(T)$ and $P_{\text{model}}(T)$ on the left and marginal distributions $P(Y)$ and $P_{\text{model}}(Y)$ on the right.



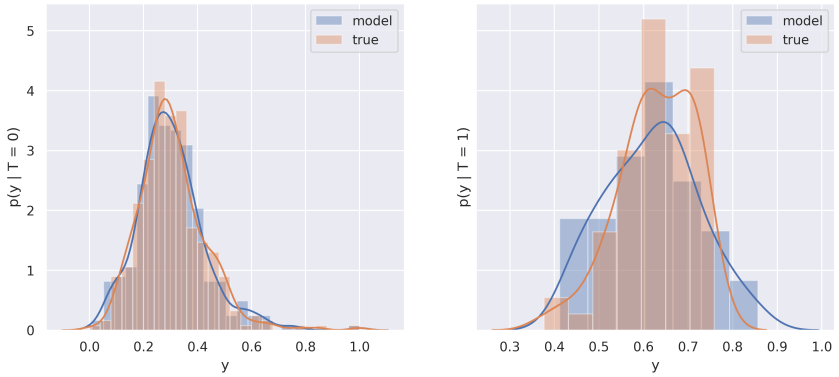
(d) Histogram and kernel density estimate visualization of $P(Y \mid T)$ and $P_{\text{model}}(Y \mid T)$. Both graphs share the same y-axis.

Figure 4: Twins – Visualizations of how well the generative model models the dataset. Figures (a) - (c) are visualizations of the sigmoidal flow model. Figures (d) - (f) are visualizations of the baseline linear Gaussian model.

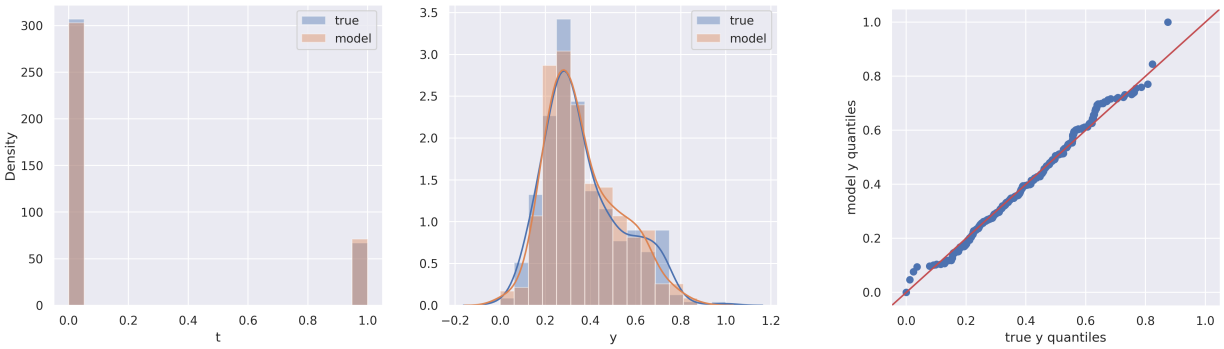


(a) Marginal distributions $P(T)$ and $P_{\text{model}}(T)$ on the left and marginal distributions $P(Y)$ and $P_{\text{model}}(Y)$ on the right.

(b) Q-Q plot of $P_{\text{model}}(Y)$ and $P(Y)$.

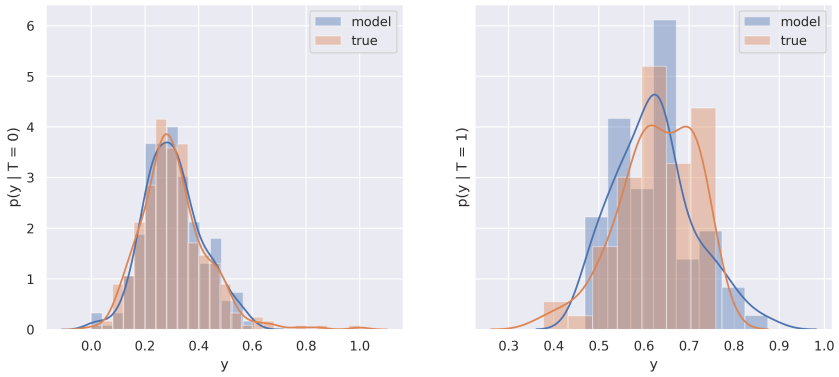


(c) Histogram and kernel density estimate visualization of $P(Y \mid T)$ and $P_{\text{model}}(Y \mid T)$. Both graphs share the same y-axis.



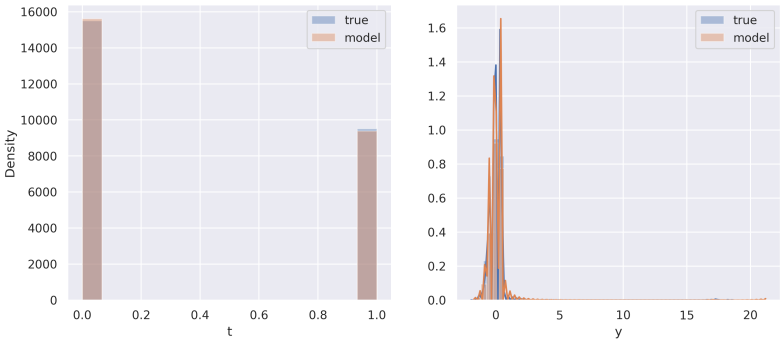
(d) Marginal distributions $P(T)$ and $P_{\text{model}}(T)$ on the left and marginal distributions $P(Y)$ and $P_{\text{model}}(Y)$ on the right.

(e) Q-Q plot of $P_{\text{model}}(Y)$ and $P(Y)$.

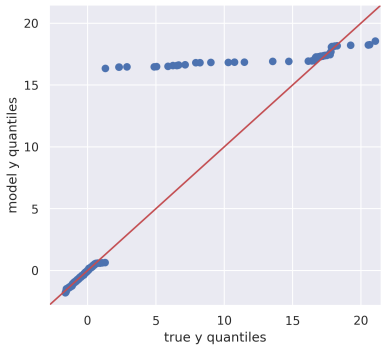


(f) Histogram and kernel density estimate visualization of $P(Y \mid T)$ and $P_{\text{model}}(Y \mid T)$. Both graphs share the same y-axis.

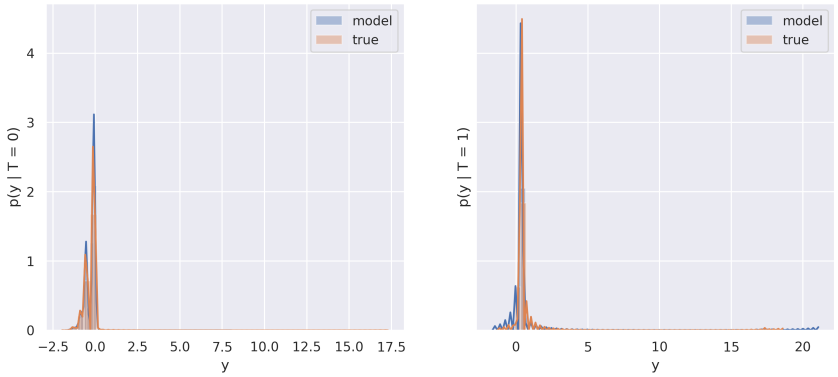
Figure 5: IHDP – Visualizations of how well the generative model models the dataset. Figures (a) - (c) are visualizations of the sigmoidal flow model. Figures (d) - (f) are visualizations of the baseline linear Gaussian model.



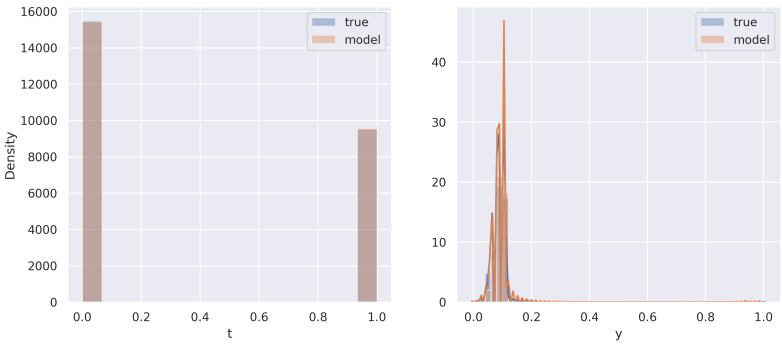
(a) Marginal distributions $P(T)$ and $P_{\text{model}}(T)$ on the left and marginal distributions $P(Y)$ and $P_{\text{model}}(Y)$ on the right.



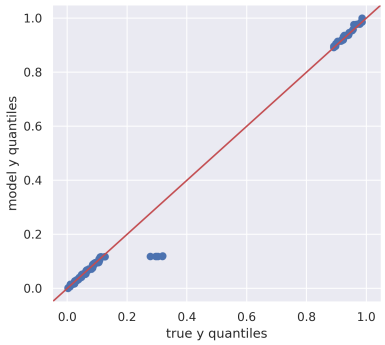
(b) Q-Q plot of $P_{\text{model}}(Y)$ and $P(Y)$.



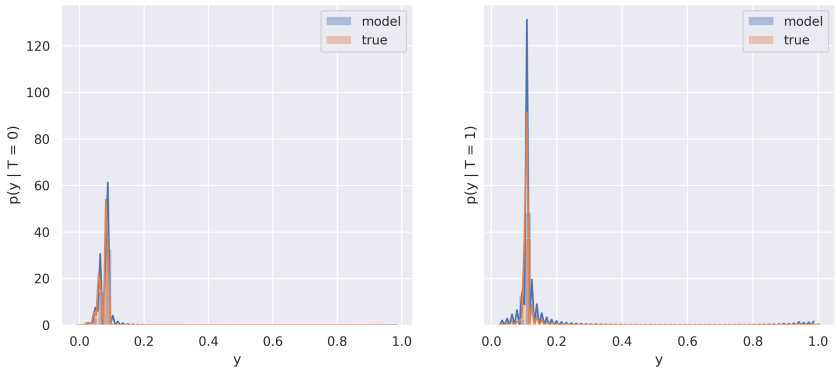
(c) Histogram and kernel density estimate visualization of $P(Y \mid T)$ and $P_{\text{model}}(Y \mid T)$. Both graphs share the same y-axis.



(d) Marginal distributions $P(T)$ and $P_{\text{model}}(T)$ on the left and marginal distributions $P(Y)$ and $P_{\text{model}}(Y)$ on the right.

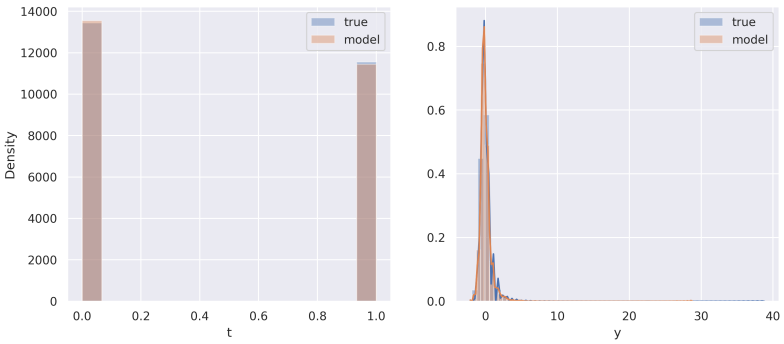


(e) Q-Q plot of $P_{\text{model}}(Y)$ and $P(Y)$.

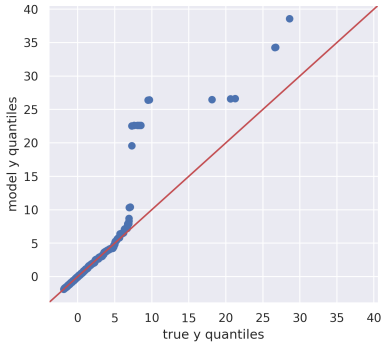


(f) Histogram and kernel density estimate visualization of $P(Y \mid T)$ and $P_{\text{model}}(Y \mid T)$. Both graphs share the same y-axis.

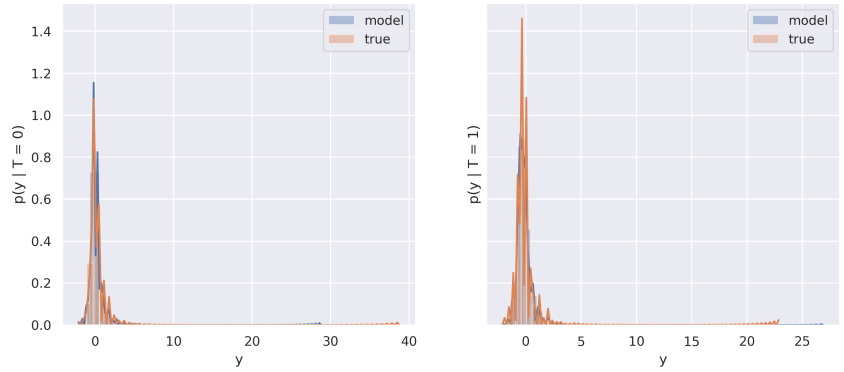
Figure 6: LBIDD-Quadratic – Visualizations of how well the generative model models the dataset. Figures (a) - (c) are visualizations of the sigmoidal flow model. Figures (d) - (f) are visualizations of the baseline linear Gaussian model.



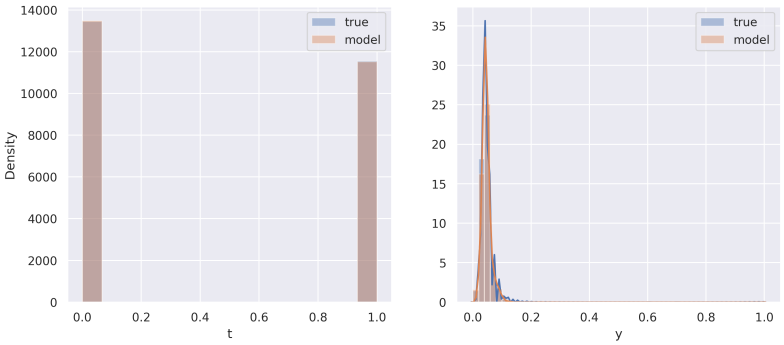
(a) Marginal distributions $P(T)$ and $P_{\text{model}}(T)$ on the left and marginal distributions $P(Y)$ and $P_{\text{model}}(Y)$ on the right.



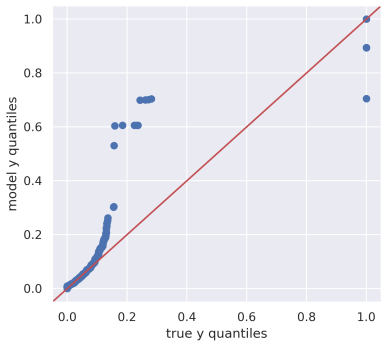
(b) Q-Q plot of $P_{\text{model}}(Y)$ and $P(Y)$.



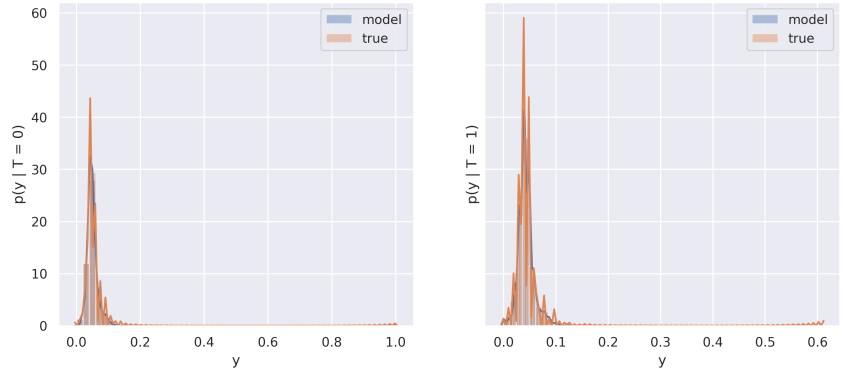
(c) Histogram and kernel density estimate visualization of $P(Y \mid T)$ and $P_{\text{model}}(Y \mid T)$. Both graphs share the same y-axis.



(d) Marginal distributions $P(T)$ and $P_{\text{model}}(T)$ on the left and marginal distributions $P(Y)$ and $P_{\text{model}}(Y)$ on the right.



(e) Q-Q plot of $P_{\text{model}}(Y)$ and $P(Y)$.



(f) Histogram and kernel density estimate visualization of $P(Y \mid T)$ and $P_{\text{model}}(Y \mid T)$. Both graphs share the same y-axis.

Figure 7: LBIDD-Exponential – Visualizations of how well the generative model models the dataset. Figures (a) - (c) are visualizations of the sigmoidal flow model. Figures (d) - (f) are visualizations of the baseline linear Gaussian model.

B. Additional Causal Estimator Benchmark Results

Table 6: Performance of inverse probability weighting (IPW) with several different propensity score models. Lower values are better in all columns except the Bias($\hat{\tau}$) column.

Propensity Score Model	Bias($\hat{\tau}$)	|Bias($\hat{\tau}$)|	Std($\hat{\tau}$)	RMSE($\hat{\tau}$)
LogisticRegression_l2	4267.47	4267.47	10698.99	11518.66
LogisticRegression_l2_trim_weights	-3128.51	3128.51	2441.88	3968.67
LogisticRegression_l2_stabilized_weights	4267.47	4267.47	10698.99	11518.66
LogisticRegression_l2_trim_stabilized_weights	-3128.51	3128.51	2441.88	3968.67
LogisticRegression	4261.71	4261.71	10723.55	11539.35
LogisticRegression_trim_weights	-3119.00	3119.00	2459.03	3971.77
LogisticRegression_stabilized_weights	4261.71	4261.71	10723.55	11539.35
LogisticRegression_trim_stabilized_weights	-3119.00	3119.00	2459.03	3971.77
LogisticRegression_l2_liblinear	1412.34	1412.34	7961.36	8085.66
LogisticRegression_l2_liblinear_trim_weights	-3955.38	3955.38	2379.15	4615.77
LogisticRegression_l2_liblinear_stabilized_weights	1412.34	1412.34	7961.36	8085.66
LogisticRegression_l2_liblinear_trim_stabilized_weights	-3955.38	3955.38	2379.15	4615.77
LogisticRegression_l1_liblinear	4785.87	4785.87	10438.06	11482.93
LogisticRegression_l1_liblinear_trim_weights	-3019.23	3019.23	2610.70	3991.43
LogisticRegression_l1_liblinear_stabilized_weights	4785.87	4785.87	10438.06	11482.93
LogisticRegression_l1_liblinear_trim_stabilized_weights	-3019.23	3019.23	2610.70	3991.43
LogisticRegression_l1_saga	4867.81	4867.81	10544.66	11614.02
LogisticRegression_l1_saga_trim_weights	-3045.84	3045.84	2586.26	3995.73
LogisticRegression_l1_saga_stabilized_weights	4867.81	4867.81	10544.66	11614.02
LogisticRegression_l1_saga_trim_stabilized_weights	-3045.84	3045.84	2586.26	3995.73
kNN	-4453.48	4453.48	1635.12	4744.17
kNN_trim_weights	-4489.71	4489.71	1635.18	4778.21
kNN_stabilized_weights	-4453.48	4453.48	1635.12	4744.17
kNN_trim_stabilized_weights	-4489.71	4489.71	1635.18	4778.21
DecisionTree	-3251.68	3251.68	3255.49	4601.27
DecisionTree_trim_weights	-4166.64	4166.64	2071.81	4653.31
DecisionTree_stabilized_weights	-3251.68	3251.68	3255.49	4601.27
DecisionTree_trim_stabilized_weights	-4166.64	4166.64	2071.81	4653.31
GaussianNB	18325.15	18325.15	15215.77	23818.71
GaussianNB_trim_weights	4220.70	4220.70	3429.33	5438.25
GaussianNB_stabilized_weights	18325.15	18325.15	15215.77	23818.71
GaussianNB_trim_stabilized_weights	4220.70	4220.70	3429.33	5438.25
QDA	19481.99	19481.99	15786.64	25075.21
QDA_trim_weights	2532.95	2532.95	2480.04	3544.91
QDA_stabilized_weights	19481.99	19481.99	15786.64	25075.21
QDA_trim_stabilized_weights	2532.95	2532.95	2480.04	3544.91