`tfb`: a Package for Targeted Function Balancing

A Thesis

Presented to

The Division of Mathematical and Natural Sciences

Reed College

In Partial Fulfillment

of the Requirements for the Degree

Bachelor of Arts

Kenai Burton-Heckman

December 2023

Approved for the Division
(Mathematics - Statistics)

Leonard Wainstein

Acknowledgements

I'd like to thank my advisor, Leonard Wainstein; my parents, Renée Burton and Christopher Heckman; and Ashley Pival, for helping me survive writing this.

List of Abbreviations

ATC	average treatment effect on the controlled
ATE	average treatment effect
ATT	average treatment effect on the treated
CEF	conditional expectation function
DAG	Directed Acyclic Graph
DGP	data generating process
DIM	difference in means
EWC	empirical weighting condition
KBAL	kernel balancing
KRLS	kernel regularized least squares
MSE	mean squared error
OLS	ordinary least squares
RCT	randomized control trial
SATC	sample average treatment effect on the controlled
SATE	sample average treatment effect
SATT	sample average treatment effect on the treated
SUTVA	stable unit treatment value assumption
TFB	targeted function balancing
WC	weighting condition
WDIM	weighted difference in means

Table of Contents

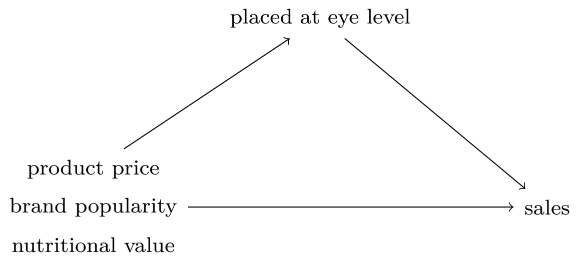
Abstract

Retrieving causal estimates from data is a worthwhile problem, but observational studies frequently make the task more difficult. When observables confound the relationship between a treatment of interest and an outcome, conditioning on the observables may satisfy conditional ignorability. The assumption is relevant to the potential outcomes framework of Splawa-Neyman, Dabrowska, & Speed (1990), and allows for potentially unbiased estimation through the identification of conditional expectation functions (CEFs). One class of estimators constructed through such identification are the weighted differences in means (WDIMs). Many weighting techniques exist, and targeted function balancing (TFB) is a novel one (Wainstein, 2022). By fitting an initial regression, the method is able to find weights that minimize a worst-case scenario of imbalance in the data within a probabilistically defined subset of functions near the initial estimate that ideally contains the true CEF of the potential outcomes. TFB transforms the problem of weighting observables into a problem of specifying the CEF correctly, creating a scalable weighting method that rewards the researcher's domain knowledge. This thesis presents a motivation of TFB for the average treatment effect (ATE), and introduces a package `tfb` for the application of the method in R, with a focus on the regression technique kernel regularized least squares (KRLS) (Hainmueller & Hazlett, 2014).

Introduction

In observational studies, there is sometimes the desire to estimate the effect of a dichotomous treatment on an outcome. For example, a grocery store owner might want to know the effect of placing a product at eye level on that product's sales, as opposed to the product being placed on a lower or higher shelf. The owner suspects that the relationship could be modeled like this, where an arrow represents a causal effect:

Figure 0.1.1



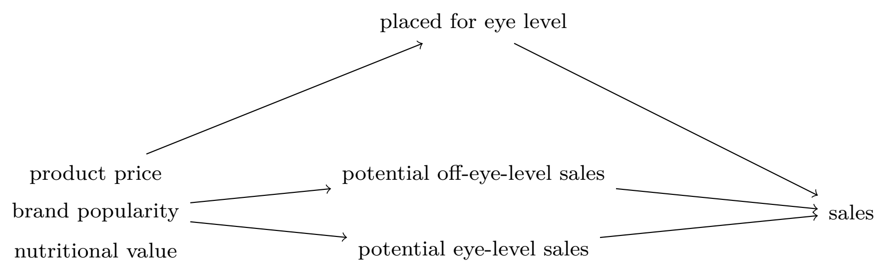
A directed acyclic graph (DAG) representing the causal structure of the effect of placing products at eye level at the grocery store on that product's sales.

Put in words, this graph is a way to represent the owner's informed beliefs. They think that placing a product at eye level will increase its sales, and they think that the product's nutrional value, popularity, and price will also impact how well it sells. However, they also believe that the product's nutritional value, popularity, and price could affect whether it gets placed at eye level or not (as suppliers are able to negotiate with the grocery store about shelf placement).

While the owner could know all the information in this graph (they might estimate brand popularity using stock prices, or some other measure of the relative "value" of a competitor), they wouldn't know how to disentangle the effect of a product being at eye level! It's not like they can run an experiment where products are randomly assigned to shelves, and the effect of a product being at eye level is confounded by all the other factors involved in how a consumer chooses a product to buy.

However, the owner isn't actually stuck in this situation. They know that if a product were to be placed out of the customer's immediate view, it would potentially have a certain amount of sales, and if it were to be placed at eye level, it would potentially have a (possibly different) amount of sales. We might then redraw the graph in figure 0.1.1 to look like

Figure 0.1.2



A directed acyclic graph (DAG) representing the causal structure of the effect of placing products at eye level at the grocery store on that product's sales, after including potential sales.

Whether or not the product is ultimately placed at eye level or not doesn't actually affect the potential sales, it only affects the final sales of the product, by determining which of the two potential sales the owner actually observes. Then conveniently, if we observe the nutritional value, brand popularity, and product price (which we can), we are able to say that whether the product is placed at eye level or not is independent of the potential sales. The values in this observational study (and analogous values in similar studies) will be referred to with more general terms. The nutritional value, the brand popularity, and the product price are the *covariates*, the shelf placement is the dichotomous *treatment*, the sales are the *outcome*, and the potential sales are the *potential outcomes*.

The setting in this example study is suitable for causal inference, a field of statistics focused on recovering causal estimates. This thesis will focus on a causal inference method called targeted function balancing (TFB), which is appropriate for this type of setting (Wainstein, 2022). TFB has been developed for the average treatment effect on the treated (ATT), one of the possible targets for estimating a causal effect (and subsequently, the average treatment effect on the controlled or ATC). This thesis expands the method to the the average treatment effect (ATE), as well as creating an R package for its implementation. TFB can be used in conjunction with many different statistical modeling techniques. Ordinary least squares (OLS) and kernel regularized least squares (KRLS) as given by Hainmueller & Hazlett (2014) are the two that have been coded thus far.

The goal of this thesis is to provide the reader with the potential to recognize possible applications of TFB and the coding tools to recover a causal estimate from that application. It will cover the necessary background for TFB, including causal inference, weighting and KRLS. Then the thesis will motivate TFB for the ATE, demonstrate its performance on simulated data, cover how to use the associated package, and conclude.

Chapter 1

Background

1.1 Notation

For a given dataset with n observations, let X_i be a 1 by p vector representing the ith observation in a given dataset with p covariates. D_i is the ith observation's treatment group, with $D_i = d \in \{0, 1\}$ representing the control and treatment groups respectively. Y_i is the ith observation's outcome. We can then have X be the n by p matrix of covariates, D be the treatment vector, and Y be the outcome vector in the study, constructed as:

$$X = \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{pmatrix} \qquad D = \begin{pmatrix} D_1 \\ D_2 \\ \vdots \\ D_n \end{pmatrix} \qquad Y = \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{pmatrix}$$

Next, let the number of control observations be n_0, and the number of treatment observations be n_1 such that $n = n_0 + n_1$. Under the potential outcomes framework, $Y_i(0)$ is the ith observation's control outcome, and $Y_i(1)$ is the ith observation's treatment outcome (Splawa-Neyman et al., 1990). D_i determines whether the value for $Y_i(0)$ or the value for $Y_i(1)$ is observed for each observation, which is aggregated into Y_i.

$$Y_i = Y_i(0) \cdot (1 - D_i) + Y_i(1) \cdot D_i$$

Then $Y(0)$ is the vector of control outcomes for all n observations in our data, and $Y(1)$ is the vector of treatment outcomes.

$$Y(0) = \begin{pmatrix} Y_1(0) \\ Y_2(0) \\ \vdots \\ Y_n(0) \end{pmatrix} \qquad Y(1) = \begin{pmatrix} Y_1(1) \\ Y_2(1) \\ \vdots \\ Y_n(1) \end{pmatrix}$$

f_d is the conditional expectation function (CEF) of $Y(d)$:

$$f_d(X) = E[Y(d) \mid X]$$

We'll say that f_d is linear in an arbitrary transformation of X, $g_d(X)$. Note that this is not much of a restriction, we could let $g_d(X) = I_n$ and f_d would be linear in the transformation regardless of the values the CEF actually takes. f_d may therefore also be represented as a p_d by 1 vector β_d, where p_d is the number of covariates or columns in $g_d(X)$. This allows for the construction of models

$$\begin{aligned} Y_i(d) &= f_d(X_i) + \epsilon_i(d) \\ &= g_d(X_i)\beta_d + \epsilon_i(d) \\ \text{with} \qquad E\left[\epsilon_i(d) \mid X_i\right] &= 0 \qquad \operatorname{var}(\epsilon_i(d) \mid X_i) = \sigma_i^2(d) \end{aligned}$$

Where $g_d(X_i)$ is the ith row of $g_d(X)$. In the potential outcomes framework, there are three important estimands: the average treatment effect (ATE), the average treatment effect on the treated (ATT), and the average treatment effect on the controlled (ATC) (Splawa-Neyman et al., 1990). They are defined as

$$\begin{aligned} \text{ATE} &= E[Y_i(1) - Y_i(0)] \\ \text{ATT} &= E[Y_i(1) - Y_i(0) \mid D_i = 1] \\ \text{ATC} &= E[Y_i(1) - Y_i(0) \mid D_i = 0] \end{aligned}$$

With the sample analogs: the sample average treatment effect (SATE), the sample average treatment effect on the treated (SATT), and the sample average treatment effect on the controlled (SATC).

$$\begin{aligned}
\text{SATE} &= \frac{1}{n}\sum_{i=1}^{n} Y_i(1) - \frac{1}{n}\sum_{i=1}^{n} Y_i(0) \\
\text{SATT} &= \frac{1}{n_1}\sum_{i:\, D_i=1} Y_i(1) - \frac{1}{n_1}\sum_{i:\, D_i=1} Y_i(0) \\
\text{SATC} &= \frac{1}{n_0}\sum_{i:\, D_i=0} Y_i(1) - \frac{1}{n_0}\sum_{i:\, D_i=0} Y_i(0)
\end{aligned}$$

1.2 Causal Inference and Potential Outcomes

In a randomized control trial (RCT), a treatment is assigned to observations at random. Randomization schemes may vary, but treatment assignment in an RCT will always be independent of the covariates. This is called ignorability, and is notated as $(Y_i(0), Y_i(1)) \perp\!\!\!\perp D_i$. This is *not* the same as saying $Y_i \perp\!\!\!\perp D_i$, as the treatment will affect which potential outcome we observe. We can then say that the two populations in the RCT are identical in the distribution of their covariates–the only empirical difference in their shape will be due to random variation. Intuitively, because we have eliminated confounding effects through random treatment assignment, we can say that the only difference between a treated observation and a control observation, on average, is the causal effect of the treatment on the outcome. This allows researchers to derive a causal estimate, a quantification of the effect of the treatment, from the data. However, RCTs are an ideal scenario.

In observational studies, where the treatment assignment mechanism is unknown, or in an experiment where the treatment assignment is dependent on covariates, researchers may be deterred from determining a causal estimate and instead focus on determining associations or predictive modeling. This is because ignorability is rarely met in observational studies. While avoiding any causal inference will inevitably be the wisest choice in some scenarios, the field provides tools for recovering a causal estimate in some others.

One of these scenarios is when we know the common causes of our treatment and our potential outcomes and observe them all in our covariates. This is the motivation for conditional ignorability, an assumption that the potential outcomes are independent of the treatment assignment *given* the covariates, or $(Y_i(0), Y_i(1)) \perp\!\!\!\perp D_i \mid X_i$. Under the potential outcomes framework, the treatment does *not* directly affect the potential outcomes, it only affects the outcome by selecting which potential outcome we observe (Splawa-Neyman et al., 1990). This means that when we condition on our covariates (assuming there is no unobserved confounding), our treatment and the potential outcomes are independent of each other, and we can recover a causal estimate. We can visualize this through a directed acyclic graph:

Figure 1.2.1

$$\begin{tikzcd}
 & X \arrow[dl] \arrow[d] \arrow[dr] & \\
D \arrow[dr] & Y(0) \arrow[d] & Y(1) \arrow[dl] \\
 & Y &
\end{tikzcd}$$

Figure 1.2.2

$$\begin{tikzcd}
 & X & \\
D \arrow[dr] & Y(0) \arrow[d] & Y(1) \arrow[dl] \\
 & Y &
\end{tikzcd}$$

DAGs representing the causal structure of the effect of a treatment on an outcome, including potential outcomes, before and after conditioning on observables.

In figure 1.2.1, we see an example of an observational study design where conditional ignorability is met. Our covariates are the only confounders of our treatment and our potential outcomes, shown by the directed edges from X to D and $(Y(0), Y(1))$ representing causal effects. In turn, the potential outcomes affect the outcome by mechanism of being observed by the treatment, shown by the directed edges from D and $(Y(0), Y(1))$ to Y. In figure 1.2.2, after conditioning on our covariates, we control the causal effect of X on D and $(Y(0), Y(1))$, shown by removing the directed edges. This allows for the independence of our treatment and our potential outcomes, represented by the conditional ignorability assumption $(Y_i(0), Y_i(1)) \perp\!\!\!\perp D_i \mid X_i$. For the purposes of TFB, we will assume that conditional ignorability holds on our data.

1.3 Weighting

A naive estimator for the ATE is the difference in means (DIM), $\hat{\tau}_{\text{dim}}$. It is defined as:

$$\hat{\tau}_{\text{dim}} = \frac{1}{n_1} \sum_{i:\, D_i=1} Y_i(1) - \frac{1}{n_0} \sum_{i:\, D_i=0} Y_i(0)$$

Unfortunately, when ignorability is not met, the DIM can become highly biased. However, with conditional ignorability, we can augment the DIM in a few different ways to be unbiased for a given estimand. This is done by introducing weights w_i for each observation. A vector of weights w is then defined as

$$w = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix}$$

The DIM can then be restructured as the weighted difference in means (WDIM), $\hat{\tau}_{\text{wdim}}$, which has a different form for each of the three estimands, respectively given by

$$\begin{aligned}
\hat{\tau}_{\text{wdim}^{\text{ATE}}} &= \frac{1}{n_1} \sum_{i:\, D_i=1} w_i Y_i(1) - \frac{1}{n_0} \sum_{i:\, D_i=0} w_i Y_i(0) \\
\hat{\tau}_{\text{wdim}^{\text{ATT}}} &= \frac{1}{n_1} \sum_{i:\, D_i=1} Y_i(1) - \frac{1}{n_0} \sum_{i:\, D_i=0} w_i Y_i(0) \\
\hat{\tau}_{\text{wdim}^{\text{ATC}}} &= \frac{1}{n_1} \sum_{i:\, D_i=1} w_i Y_i(1) - \frac{1}{n_0} \sum_{i:\, D_i=0} Y_i(0)
\end{aligned}$$

The thought is that, if the two distributions of the covariates are made similar enough by weighting, we can expect the control units, if they were to be treated, to have a similar distribution of potential outcomes to the treatment units (and similarly, the control outcomes for the treated group would reflect the outcomes of the control group). For example, for the ATT:

$$\begin{aligned}
\text{ATT} &= E[Y_i(1) - Y_i(0) \mid D_i = 1] \\
&= E[Y_i(1) \mid D_i = 1] - E[Y_i(0) \mid D_i = 1] \qquad \text{linearity of expectation}
\end{aligned}$$

We see that the difficult portion of the estimand to estimate is $E[Y_i(0) \mid D_i = 1]$, as we have no information about the control potential outcomes within the treated group. What we *can* do is use the law of iterated expectation:

$$E[Y_i(0) \mid D_i = 1] = E\left[E[Y_i(0) \mid X_i] \mid D_i = 1\right]$$

Inspecting this CEF, we see that the inside expectation is a function of X_i, and that the outside expectation is conditional on $D_i = 1$. This means that we now have the CEF over the probability distribution $p(X_i \mid D_i = 1)$, a function that we can empirically estimate with the distribution of our covariates in the treated group. We don't know what the CEF actually is, but we will use our knowledge of its probability distribution.

We would like for our naive $\hat{\tau}_{\text{dim}}$ to be unbiased for the ATT, but this would require the expected value of the DIM to equal the ATT, and we can see that this is not the case.

$$\begin{aligned}
E[\hat{\tau}_{\text{dim}}] &= E\left[\frac{1}{n_1}\sum_{i:\, D_i=1} Y_i(1) - \frac{1}{n_0}\sum_{i:\, D_i=0} Y_i(0)\right] \\
&= E\left[\frac{1}{n_1}\sum_{i:\, D_i=1} Y_i(1)\right] - E\left[\frac{1}{n_0}\sum_{i:\, D_i=0} Y_i(0)\right] && \text{linearity of expectation} \\
&= E\left[Y_i(1) \mid D_i = 1\right] - E\left[Y_i(0) \mid D_i = 0\right] \\
&= E\left[Y_i(1) \mid D_i = 1\right] - E\left[E[Y_i(0) \mid X_i] \mid D_i = 0\right] && \text{law of iterated expectation}
\end{aligned}$$

Fortunately we do know the rightmost CEF, and we know that its probability distribution is over $p(X_i \mid D_i = 0)$. This is what motivates our weighted difference in means. If we can weight the distribution of covariates in the control group to be the same as the desired distribution of covariates in the treatment group, our estimator will be unbiased for the ATT. So the weighted difference in means for the ATT ideally has weights such that $w_i p(X_i \mid D_i = 0)$ approximates $p(X_i \mid D_i = 1)$. Similarly, the WDIM for the ATC shifts the distribution of covariates in the treatment group towards the desired distribution of covariates in the control group, and the WDIM for the ATE shifts both the covariate distribution in the control group and the covariate distribution in the treated group towards the desired covariate distribution unconditional on D_i.

There are multiple methods for determining these weights. For example, in inverse propensity score weighting, we attempt to equate the distributions of covariates between the treatment and control groups using the propensity score, the probability of being treated given the covariates (Rosenbaum & Rubin, 1983). However, this method is heavily biased by misspecification and suffers from the curse of dimensionality, frequently making it unrealistic. As a result, methods of balancing weights have also been developed, which seek to equate the means of the covariates between the treatment and control groups instead. We define the *imbalance* in X, varying by estimand, as

$$\begin{aligned}
\text{imbal}_d^{\text{ATE}}(w, X, D) &= \frac{1}{n}\sum_{i=1}^{n} X_i - \frac{1}{n_d}\sum_{i:\, D_i=d} w_i X_i \\
\text{imbal}^{\text{ATT}}(w, X, D) &= \frac{1}{n_1}\sum_{i:\, D_i=1} X_i - \frac{1}{n_0}\sum_{i:\, D_i=0} w_i X_i \\
\text{imbal}^{\text{ATC}}(w, X, D) &= \frac{1}{n_1}\sum_{i:\, D_i=1} w_i X_i - \frac{1}{n_0}\sum_{i:\, D_i=0} X_i
\end{aligned}$$

Then these methods of balancing weights tend to make the imbalance in X zero, or very small. Note that the imbalance for the ATE comes in two parts because we have two targets when making the WDIM unbiased: approximating the distribution of X_i with the distribution of $X_i \mid D_i = 0$, and approximating the distribution of X_i with the distribution of $X_i \mid D_i = 1$.

Exact balancing weights, such as entropy balancing, force the imbalance in covariates to be zero (Hainmueller, 2012). However, exact balance is not always feasible, especially in high-dimensional spaces. Approximate balancing weights, such as kernel balancing, instead seek approximate balance on the covariates, making the imbalance close to zero without requiring it to be zero (Hazlett, 2020). Targeted Function Balancing, or TFB, is a novel method of approximate balancing weights (Wainstein, 2022). In addition to seeking balance on the covariates, TFB further aims for: balancing an initial regression on the data notated $\hat{f}_d(X)$ that estimates $f_d(X)$, prioritizing balance in covariates with high variance coefficients in an estimate of β_d that we notate $\hat{\beta}_d$, and to minimize the variance of its weights. This is analogous to a penalized regression: minimizing the imbalance in covariates is only one priority, similar to how minimizing the sum of squared residuals is only one priority in ridge regression (Hoerl & Kennard, 2000). In the context of TFB, we might think of our weights being penalized by their variance as "shrinking" them towards one, similarly to how the coefficients in ridge regression are shrunk towards zero (although the variance of the weights is technically not regularized by a parameter).

1.4 Assumptions

The assumptions and conditions are formally presented here for the sake of having them in one place, and because they will be used throughout the thesis. However, they are rather technical and not all of the discussion presented here is immediately necessary. The reader may instead find it helpful to skim them and refer back to read them more thoroughly as they become more relevant in the text.

1. Dichotomous Treatment

   While causal inference methods for continuous treatment effects do exist, TFB has not yet been generalized beyond a dichotomous treatment. This extension of the method is being considered, however–see the section on further work.

2. Stable Unit Treatment Value Assumption (SUTVA)

   This assumption is built into the potential outcomes framework, and therefore won't be discussed outside of this section (Splawa-Neyman et al., 1990). It requires two things. The first is that the potential outcomes for each observation are independent of the treatment status of other observations. If this part of the SUTVA is violated, the number of potential outcomes explodes to 2^n. The second is that if two observations receive treatment, they receive the same treatment. For example, the setting presented in the introduction would violate this second part of the SUTVA. A product being at eye level or not is presented dichotomously, but it would be reasonable to expect that the potential outcomes for a product would change if it was placed on a higher shelf relative to eye level instead of on a lower shelf.

3. Conditional Ignorability

   Conditional ignorability, $(Y_i(1), Y_i(0)) \perp\!\!\!\perp D_i \mid X_i$, is required for the recovery of a causal estimate when ignorability is not met. This means that the potential outcomes need to be independent of the treatment after conditioning on X. This assumption is near-impossible to guarantee in practice, since conditional ignorability implies that there is no unobserved confounding between D and Y.

4. Distribution of Error

   We assume that the error terms $\epsilon_i(d)$ are expectation zero with finite variance. The researcher can attempt to check this assumption by looking at the distribution of residuals in $\hat{f}_d(X)$. They should be centered at zero.

5. Weights Sum to n_d

   We want to choose weights that sum to n_d, so we will assume this property in our motivation and impose it as a restriction on our function. Specifically, weights will have

$$
\begin{aligned}
E\left[w_i\right] &= 1 \\
\operatorname{var}(w_i) &= E[w_i^2] - 1
\end{aligned}
$$

   And so we will generally think of $\frac{1}{n}\sum_{i=1}^{n} w_i^2$ as the variance of the weights, because the sample mean is unbiased for the expectation $E[w_i^2]$ (and we can't affect the constant portion of $\operatorname{var}(w_i)$ through our choice in weights). We then have assumptions

$$
\begin{aligned}
\sum_{i:\, D_i=0} w_i &= n_0 \\
\sum_{i:\, D_i=1} w_i &= n_1 \\
\sum_{i=1}^{n} w_i &= n \\
\sum_{i:\, D_i=0} w_i + \sum_{i:\, D_i=1} w_i &= n_0 + n_1
\end{aligned}
$$

   Where we trivially have $w_i = 1$ for any unweighted observation.

6. Linearity of f_d in g_d

   We assume that the true functional form of the relationship between Y and $g_d(X)$ has a linear representation, f_d. This will end up being important for the derivation of TFB as a problem. However this isn't particularly necessary for the method, as TFB can be generalized to nonlinear regression functions. $g_d(X)$ has to have the same number of rows as X and at least one covariate, but its shape can otherwise vary.

1.5 Conditions

1. Empirical Weighting Conditions (EWCs)

   In order for $\hat{\tau}_{\text{wdim}}$ to be a consistent estimator of a given estimand, we want the respective weighting conditions (WCs) to hold:

$$\begin{aligned}
&E[w_iY_i(d) \mid D_i = d] = E[Y_i(d)] && \text{WC-ATE}\\
&E[w_iY_i(0) \mid D_i = 0] = E[Y_i(0) \mid D_i = 1] && \text{WC-ATT}\\
&E[w_iY_i(1) \mid D_i = 1] = E[Y_i(1) \mid D_i = 0] && \text{WC-ATC}
\end{aligned}$$

We won't assume this, but TFB will attempt to find weights such that their empirical analogs (the EWCs) hold. This means that TFB will find weights that satisfy the WCs in expectation. The EWCs are in terms of the imbalances that have already been defined, but do note the transformation of X by f_d. We define the EWCs as

$$\begin{aligned}
\text{imbal}_d^{\text{ATE}}(w, f_d(X), D) &= \frac{1}{n}\sum_{i=1}^{n} f_d(X_i) - \frac{1}{n_d}\sum_{i:\ D_i=d} w_i f_d(X_i)\\
&= 0 && \text{EWC-ATE}\\
\text{imbal}^{\text{ATT}}(w, f_0(X), D) &= \frac{1}{n_1}\sum_{i:\ D_i=1} f_0(X_i) - \frac{1}{n_0}\sum_{i:\ D_i=0} w_i f_0(X_i)\\
&= 0 && \text{EWC-ATT}\\
\text{imbal}^{\text{ATC}}(w, f_1(X), D) &= \frac{1}{n_1}\sum_{i:\ D_i=1} w_i f_1(X_i) - \frac{1}{n_0}\sum_{i:\ D_i=0} f_1(X_i)\\
&= 0 && \text{EWC-ATC}
\end{aligned}$$

And the EWC biases as

$$\begin{aligned}
\text{EWC Bias}^{\text{ATE}} &= \text{imbal}_0^{\text{ATE}}(w, f_0(X), D) - \text{imbal}_1^{\text{ATE}}(w, f_1(X), D)\\
\text{EWC Bias}^{\text{ATT}} &= \text{imbal}^{\text{ATT}}(w, f_0(X), D)\\
\text{EWC Bias}^{\text{ATC}} &= \text{imbal}^{\text{ATC}}(w, f_1(X), D)
\end{aligned}$$

2. Asymptotic Normality or Normality of $\hat{\beta}_d$

While TFB can be motivated without this condition and still give an estimator with desirable properties, in order for TFB's *motivation* to be a *derivation*, we would like this to hold. When it comes to the possible forms of $\hat{f}_d$ discussed in this thesis, the asymptotic normality of OLS's $\hat{\beta}_d$ has been established (Grenander, 1954). KRLS's $\hat{\beta}_d$ is not asymptotically normal, but we will show that it is normal in the motivation section if we assume the error terms $\epsilon_i(d)$ are normal. Note that this is stronger than TFB's assumption that error terms are expectation zero with finite variance. I suspect that this condition can actually be relaxed to only require that the predicted outcome $\hat{f}_d(X_i)$ is asymptotically normal–see the motivation section for discussion of this. If I am correct, then we do not need to assume the normality of error for KRLS, as the asymptotic normality of its predicted outcomes has been proven under the weaker assumption that error terms are expectation zero with finite variance in the appendix of Hainmueller & Hazlett (2014).

3. (Working Condition) Proper specification of f_d

   While TFB is robust against the misspecification of the true functional relationship between Y and $g_d(X)$ to an extent, it is likely that f_d must be specified by $\hat{f}_d$ within a certain degree of proximity in order for unbiased estimation to be possible. This asymptotic property has not been proved yet–see the section on further work. For now, an individual wanting to use TFB in practice can give this condition proper consideration by choosing an initial regression function appropriate for their data. For example, a particularly nonlinear relationship might be best modeled by KRLS (Hainmueller & Hazlett, 2014).

4. (Working Condition) Cross-Fitting

   Cross-fitting TFB is likely required in order for $\hat{\tau}_{\text{wdim}}$ to be asymptotically normal. This can be done by k-fold sample splitting the data uniformly at random, fitting initial regressions on each fold, weighting observations within a fold using a derangement of those regressions, and reporting the median of the WDIMs generated by each fold as the final estimate. $k = 2$ folds are used throughout this thesis and in the `tfb` package (effectively making this an average of the two estimates), though further functionality for varying k is planned. Specific choices of $k \geq 2$ may be preferable for finite sample sizes but have no effect on the asymptotics of the estimator, and they always reduce overfitting bias compared to the same method without cross-fitting (Chernozhukov et al., 2018).

1.6 Targeted Function Balancing

TFB aims to find weights that minimize the mean squared error (MSE) between the weighted difference in means and the sample estimand. By conducting an initial regression on the data, TFB can construct a set of possible β_d in $\mathbb{R}^{p_d}$ around this initial regression, representing where we probabilistically expect the true functional form of the relationship between Y and $g_d(X)$ to lie. TFB then optimizes for the β_d in this set that maximize the bias in the EWC, finding a worst-case scenario for imbalance. At this worst-case scenario, TFB then minimizes for this approximation of the MSE, in the process minimizing four things: the imbalance in the covariates, the imbalance in the predicted outcomes, the imbalance in the covariates scaled by $\hat{\beta}_d$'s covariance, and the variance of the weights. The weights that result in this minimum can then be used to find the causal estimate, $\hat{\tau}_{\text{wdim}}$.

Note that for the final causal estimate, TFB actually splits the data uniformly at random and cross-fits by finding weights and a WDIM within each sample split based on the initial regression from the other split. The mean of the two estimates is then reported as the final estimate. Cross-fitting reduces overfitting bias in the estimator as described in Chernozhukov et al. (2018). TFB would otherwise introduce said bias by optimizing for weights in the same sample that was used to optimize an initial regression.

To better understand what TFB does, we'll look at the optimization problem for the ATT,

$$\begin{aligned}
&\operatorname{argmin}_w \left[E\left[(\hat{\tau}_{\text{wdim}^{\text{ATT}}} - SATT)^2 \mid X, D\right]\right] \approx \\
&\operatorname{argmin}_w \Bigg[\Bigg(\sqrt{Q_q(\chi^2_p)} \cdot \left\|\hat{V}_{\beta_0}^{\frac{1}{2}} \text{imbal}_{\text{ATT}}(w, g_0(X), D)\right\|_2 + \\
&\qquad \left|\text{imbal}_{\text{ATT}}(w, g_0(X), D)^T \hat{\beta}_0\right|\Bigg)^2 + \frac{\hat{\sigma}_0^2}{n_0^2} \sum_{i:\, D_i=0} w_i^2\Bigg]
\end{aligned}$$

where $\hat{\sigma}_0^2$ is an estimate of $\sigma_i^2(0)$. $\hat{V}_{\beta_0}$ is the estimate of the covariance matrix of β_0, and could be equivalently represented as $V_{\hat{\beta}_0}$, the covariance matrix of $\hat{\beta}_0$.

There are three components to this optimization problem to keep track of. The first two are casually referred to as the raw imbalance and the transformed imbalance, respectively:

$$\begin{aligned}
\left|\text{imbal}_{\text{ATT}}(w, g_0(X), D)^T \hat{\beta}_0\right| &\qquad \text{component 1} \\
\sqrt{Q_q(\chi^2_p)} \cdot \left\|\hat{V}_{\beta_0}^{\frac{1}{2}} \text{imbal}_{\text{ATT}}(w, g_0(X), D)\right\|_2 &\qquad \text{component 2}
\end{aligned}$$

These two components represent the balance in the predicted outcomes and the balance in the covariates scaled by $\hat{\beta}_d$'s covariance, respectively. Their sum is an approximation of the EWC bias–this is the worst-case that we are minimizing over. Balance in the covariates also happens to imply balance in these two components, meaning they represent a total three of the four optimization targets listed at the beginning of this section. The final target (the variance of the weights) is represented by the final component:

$$\frac{\hat{\sigma}_0^2}{n_0^2} \sum_{i:\, D_i=0} w_i^2 \qquad \text{component 3}$$

This pattern of components and their associated optimization targets will show up again when motivating TFB for the ATE. Keep in mind that minimizing the mean squared error between the weighted difference in means and the sample estimand is only the motivating problem for TFB, so while our weights do ideally minimize this quantity as well, it is not truly one of the targets for optimization.

1.7 Supplementary Background on KRLS

While any initial regression that has a linear representation in an arbitrary transformation of the covariates could be used for TFB, in the scope of this paper focus will be given to ordinary least squares (OLS) and kernel regularized least squares (KRLS) (Hainmueller & Hazlett, 2014). Additional description of OLS won't be given, but the reader is less likely to be familiar with the more complex KRLS, so a light treatment of it will be given here.

One way to motivate KRLS is through the idea of observations in our data being similar to each other. If our observations are independent and identically distributed, we would expect the outcomes of observations i, j with $X_i = X_j$ to only vary in their error terms $\epsilon_d(X_i)$, implying that their outcomes should be identical in expectation. So instead of creating a linear model on the observations themselves (as in OLS), we might consider making a model on the similarity between observations.

Consider the unstandardized gaussian curve $e^{-\frac{\|x'-x\|_2^2}{\delta^2}}$, where x is a reference point (or equivalently, observation) in $\mathbb{R}^p$, x' is an arbitrary point in $\mathbb{R}^p$, and $\delta \in \mathbb{R}$ is a nonzero parameter of the model. Note that letting $x' = x$ maximizes this curve to $e^{-0} = 1$, where there is no distance between our points and they are therefore identical. And the infimum of the curve is $\lim_{\|x'-x\|\to\infty} e^{-\frac{\|x'-x\|_2^2}{\delta^2}} = 0$, so as our points get farther and farther apart, our curve trends to zero. Finally, the standard choice for δ^2 (once data has been standardized) is p, as it consistently gives a wide range of possible similarities within the data. This gaussian function then makes for a suitable measure of similarity.

A positive semi-definite kernel function is a function $k(x_i, x_j)$ such that if we construct a kernel matrix from our n observations

$$K = \begin{pmatrix} k(x_1, x_1) & \dots & k(x_1, x_n) \\ \vdots & & \vdots \\ k(x_n, x_1) & \dots & k(x_n, x_n) \end{pmatrix}$$

Then $c^T Kc \geq 0$ for nonzero $c \in \mathbb{R}^n$. While any positive semi-definite kernel function can be used for KRLS, the gaussian kernel that we already discussed as a measure of similarity is the favored choice for the method. Let y be an n by 1 response vector (this will become Y in the context of TFB). We will actually consider Kc as the space of functions that the true functional form of y can fall within, such that we observe a noisy

$$y = Kc + \epsilon$$
$$y_i = K_i c + \epsilon_i$$

where K_i is the ith row of the kernel matrix K. Note then that a single observation's outcome is a linear combination of gaussians, a smooth and highly flexible space of functions. It is so flexible that we could almost always find a $\hat{c}$ such that $\hat{c} = K^{-1}y$, as K is square and positive semi-definite by definition. Specifically, K is always invertible when no $x_i = x_j$ for $i \neq j$. Perfectly fitting $\hat{c}$ to the data would inevitably lead to overfitting, however, as we cannot be certain of the magnitude of the error terms in practice.

So to choose a more reasonable $\hat{c}$, we will introduce a regularization term. An ideal choice of $\hat{c}$ will then prefer two things. The first preference is for coefficient vectors that minimize a squared loss function, thereby looking for a good fit on our data. This is a very common target for optimization–this is the "least squares" in OLS, for example. The other preference is for low-complexity functions, which is also intuitive. There are many possible functions that can fit our data well, so choosing one that is only as complex as necessary is part of the fundamental bias-variance tradeoff in statistical modeling. Choosing a more complex, flexible function to model our data will decrease the bias of our estimator, but also increase its variance as it tries to account for the inherent noise present.

This regularizer will be $\lambda c^T Kc$ for a regularization parameter $\lambda \in \mathbb{R}$. The regularizer effectively penalizes the magnitude or complexity of our estimated functional form $K\hat{c}$. λ is typically chosen through cross-validation, like other regularization parameters. This makes the final statement of the model's form

$$\begin{aligned} \hat{y} &= K\hat{c} && \text{for} \\ \hat{c} &= \text{argmin}_{c \in \mathbb{R}^n} (y - Kc)^T (y - Kc) + \lambda c^T K c \\ &= (K + \lambda I_n)^{-1} y \end{aligned}$$

where we can make predictions for some new observation x' with

$$\hat{y}_i(x') = \begin{pmatrix} k(x', x_1) & \dots & k(x', x_n) \end{pmatrix} \hat{c} \qquad x \in \mathbb{R}^p$$

In summary, KRLS is a regularized method of least squares, fit on the gaussian kernel matrix of our data rather than directly on the data itself. KRLS is highly flexible, while it being regularized helps prevent overfitting. This makes the method suitable to social science contexts, where the true functional form is frequently suspected to be relatively smooth. Even when the true functional form is linear (making OLS the most efficient estimator), KRLS maintains usefulness by being robust against high-leverage points and overfitting.

Within the context of this paper, KRLS has an (abstracted) linear representation and is therefore viable as a regression method for TFB to produce weights for. KRLS also motivates the generalization of $f(X)$ for TFB from only linear forms to arbitrary forms through the use of $g_d(X)$, as g_d is necessarily not the identity transformation when using KRLS. Rather, we will let $g_d(X)$ be a subset of K.

Chapter 2

Motivation of the TFB optimization problem for the ATE

2.1 Rewriting the MSE

The ATE optimization problem for TFB is slightly more complex than that of the ATT or the ATC, as we have double the number of things to keep track of. However, it starts from the same motivating problem of minimizing the conditional MSE of a $\hat{\tau}_{\text{wdim}}$ as an estimate of the ATE. By definition:

$$
\begin{aligned}
E\left[(\hat{\tau}_{\text{wdim}^{\text{ATE}}} - \text{SATE})^2 \mid X, D\right] =& E\Bigg[\Bigg(\Bigg(\frac{1}{n_1}\sum_{i:\, D_i=1} w_i Y_i(1) - \frac{1}{n_0}\sum_{i:\, D_i=0} w_i Y_i(0)\Bigg) \\
& - \left(\frac{1}{n}\sum_{i=1}^{n} Y_i(1) - \frac{1}{n}\sum_{i=1}^{n} Y_i(0)\right)\Bigg)^2 \mid X, D\Bigg]
\end{aligned}
$$

We then substitute in our potential outcome models $Y_i(d) = f_d(X_i) + \epsilon_i(d)$.

$$
\begin{aligned}
= E\Bigg[\Bigg(&\left(\frac{1}{n_1}\sum_{i:\, D_i=1} w_i f_1(X_i) - \frac{1}{n_0}\sum_{i:\, D_i=0} w_i f_0(X_i)\right) - \left(\frac{1}{n}\sum_{i=1}^{n} f_1(X_i) - \frac{1}{n}\sum_{i=1}^{n} f_0(X_i)\right) \\
&+ \left(\frac{1}{n_1}\sum_{i:\, D_i=1} w_i \epsilon_i(1) - \frac{1}{n_0}\sum_{i:\, D_i=0} w_i \epsilon_i(0)\right) - \left(\frac{1}{n}\sum_{i=1}^{n} \epsilon_i(1) - \frac{1}{n}\sum_{i=1}^{n} \epsilon_i(0)\right)\Bigg)^2 \mid X, D\Bigg]
\end{aligned}
$$

We group like terms.

$$= E\Bigg[\Bigg(\left(\left(\frac{1}{n}\sum_{i=1}^{n} f_0(X_i) - \frac{1}{n_0}\sum_{i:\ D_i=0} w_i f_0(X_i)\right) - \left(\frac{1}{n}\sum_{i=1}^{n} f_1(X_i) - \frac{1}{n_1}\sum_{i:\ D_i=1} w_i f_1(X_i)\right)\right.$$
$$\left. + \left(\frac{1}{n_1}\sum_{i:\ D_i=1} w_i\epsilon_i(1) - \frac{1}{n_0}\sum_{i:\ D_i=0} w_i\epsilon_i(0)\right) - \left(\frac{1}{n}\sum_{i=1}^{n}\epsilon_i(1) - \frac{1}{n}\sum_{i=1}^{n}\epsilon_i(0)\right)\right)^2 \mid X, D\Bigg]$$

Then, by the definition of the imbalance and the EWC bias:

$$= E\Bigg[\Bigg(\left(\text{imbal}_0^{\text{ATE}}(w, f_0(X), D) - \text{imbal}_1^{\text{ATE}}(w, f_1(X), D)\right)$$
$$+ \left(\frac{1}{n_1}\sum_{i:\ D_i=1} w_i\epsilon_i(1) - \frac{1}{n_0}\sum_{i:\ D_i=0} w_i\epsilon_i(0)\right) - \left(\frac{1}{n}\sum_{i=1}^{n}\epsilon_i(1) - \frac{1}{n}\sum_{i=1}^{n}\epsilon_i(0)\right)\Bigg)^2 \mid X, D\Bigg]$$
$$= E\Bigg[\Bigg(\left(\text{EWC Bias}^{\text{ATE}}\right)$$
$$+ \left(\frac{1}{n_1}\sum_{i:\ D_i=1} w_i\epsilon_i(1) - \frac{1}{n_0}\sum_{i:\ D_i=0} w_i\epsilon_i(0)\right) - \left(\frac{1}{n}\sum_{i=1}^{n}\epsilon_i(1) - \frac{1}{n}\sum_{i=1}^{n}\epsilon_i(0)\right)\Bigg)^2 \mid X, D\Bigg]$$

Next, let's distribute the squared term. We'll be able to ignore most of it, fortunately.

$$\text{argmin}_w E\Bigg[\Bigg(\left(\text{EWC Bias}^{\text{ATE}}\right)$$
$$+ \left(\frac{1}{n_1}\sum_{i:\ D_i=1} w_i\epsilon_i(1) - \frac{1}{n_0}\sum_{i:\ D_i=0} w_i\epsilon_i(0)\right) - \left(\frac{1}{n}\sum_{i=1}^{n}\epsilon_i(1) - \frac{1}{n}\sum_{i=1}^{n}\epsilon_i(0)\right)\Bigg)^2 \mid X, D\Bigg]$$
$$= \text{argmin}_w \Bigg($$
$$\text{term 1} \quad E\left[\left(\text{EWC Bias}^{\text{ATE}}\right)^2 \mid X, D\right]$$
$$\text{term 2} \quad + 2E\left[\left(\text{EWC Bias}^{\text{ATE}}\right)\left(\frac{1}{n_1}\sum_{i:\ D_i=1} w_i\epsilon_i(1) - \frac{1}{n_0}\sum_{i:\ D_i=0} w_i\epsilon_i(0)\right) \mid X, D\right]$$
$$\text{term 3} \quad - 2E\left[\left(\text{EWC Bias}^{\text{ATE}}\right)\left(\frac{1}{n}\sum_{i=1}^{n}\epsilon_i(1) - \frac{1}{n}\sum_{i=1}^{n}\epsilon_i(0)\right) \mid X, D\right]$$
$$\text{term 4} \quad + E\left[\left(\frac{1}{n_1}\sum_{i:\ D_i=1} w_i\epsilon_i(1) - \frac{1}{n_0}\sum_{i:\ D_i=0} w_i\epsilon_i(0)\right)^2 \mid X, D\right]$$
$$\text{term 5} \quad - 2E\left[\left(\frac{1}{n_1}\sum_{i:\ D_i=1} w_i\epsilon_i(1) - \frac{1}{n_0}\sum_{i:\ D_i=0} w_i\epsilon_i(0)\right)\left(\frac{1}{n}\sum_{i=1}^{n}\epsilon_i(1) - \frac{1}{n}\sum_{i=1}^{n}\epsilon_i(0)\right) \mid X, D\right]$$
$$\text{term 6} \quad + E\left[\left(\frac{1}{n}\sum_{i=1}^{n}\epsilon_i(1) - \frac{1}{n}\sum_{i=1}^{n}\epsilon_i(0)\right)^2 \mid X, D\right]\Bigg)$$

Term 1 must be kept in, but terms 2 and 3 can be removed, because the error terms have expectation zero given X, D. Term 4 must also be kept in, and we'll leave in term 5 for now. In term 6, the unweighted error terms cannot be removed from the equation, but they can be safely ignored. In the context of finding the weights that will minimize this expression, the unweighted error terms are invariant relative to the weights, so the minimum of an expression with term 6 will use the same weights as the minimum of an expression without term 6. This leaves us with

$$\begin{aligned} = \operatorname{argmin}_w \Bigg(& E\left[\left(\text{EWC Bias}^{\text{ATE}}\right)^2 \mid X, D\right] + E\left[\frac{1}{n_1^2}\sum_{i:\ D_i=1} w_i^2\epsilon_i^2(1) + \frac{1}{n_0^2}\sum_{i:\ D_i=0} w_i^2\epsilon_i^2(0) \mid X, D\right] \\ & - 2E\left[\frac{1}{n_1}\sum_{i:\ D_i=1} w_i\epsilon_i^2(1) + \frac{1}{n_0}\sum_{i:\ D_i=0} w_i\epsilon_i^2(0) \mid X, D\right]\Bigg) \end{aligned}$$

The expectation of the difference in imbalance squared (the EWC bias) is a function of X and D, which we have conditioned on, so we are able to treat them as constant. The weights are deterministic, as we are looking to find w to minimize this expression. We can therefore remove its expected value.

$$\begin{aligned} = \operatorname{argmin}_w \Bigg(& \left(\text{EWC Bias}^{\text{ATE}}\right)^2 + E\left[\frac{1}{n_1^2}\sum_{i:\ D_i=1} w_i^2\epsilon_i^2(1) + \frac{1}{n_0^2}\sum_{i:\ D_i=0} w_i^2\epsilon_i^2(0) \mid X, D\right] \\ & - 2E\left[\frac{1}{n_1}\sum_{i:\ D_i=1} w_i\epsilon_i^2(1) + \frac{1}{n_0}\sum_{i:\ D_i=0} w_i\epsilon_i^2(0) \mid X, D\right]\Bigg) \end{aligned}$$

Squares of real numbers are nonnegative, so we can introduce absolute value around the imbalance term.

$$\begin{aligned} = \operatorname{argmin}_w \Bigg(& \left(|\text{EWC Bias}^{\text{ATE}}|\right)^2 + E\left[\frac{1}{n_1^2}\sum_{i:\ D_i=1} w_i^2\epsilon_i^2(1) + \frac{1}{n_0^2}\sum_{i:\ D_i=0} w_i^2\epsilon_i^2(0) \mid X, D\right] \\ & - 2E\left[\frac{1}{n_1}\sum_{i:\ D_i=1} w_i\epsilon_i^2(1) + \frac{1}{n_0}\sum_{i:\ D_i=0} w_i\epsilon_i^2(0) \mid X, D\right]\Bigg) \end{aligned}$$

By the definition of the error in our models, we note that

$$\begin{aligned} \sigma_i^2(d) &= \operatorname{var}(\epsilon_i(d) \mid X_i) \\ &= E[\epsilon_i^2(d) \mid X_i] - E[\epsilon_i(d) \mid X_i]^2 && \text{definition of variance} \\ &= E[\epsilon_i^2(d) \mid X_i] && E[\epsilon_i(d) \mid X_i] = 0 \end{aligned}$$

And so we can remove the expectation from the error terms as well.

$$\begin{aligned} = \operatorname{argmin}_w \Bigg(& \left(|\text{EWC Bias}^{\text{ATE}}|\right)^2 + \left(\frac{1}{n_1^2}\sum_{i:\ D_i=1} w_i^2\sigma_i^2(1) + \frac{1}{n_0^2}\sum_{i:\ D_i=0} w_i^2\sigma_i^2(0)\right) \\ & - 2\left(\frac{1}{n_1}\sum_{i:\ D_i=1} w_i\sigma_i^2(1) + \frac{1}{n_0}\sum_{i:\ D_i=0} w_i\sigma_i^2(0)\right)\Bigg) \end{aligned}$$

We don't know the true values of $\sigma_i^2(d)$, so we approximate the expression with our constant estimates $\hat{\sigma}_d^2$:

$$\begin{aligned}\approx \operatorname{argmin}_w \Bigg(&\left(|\text{EWC Bias}^{\text{ATE}}|\right)^2 + \left(\frac{\hat{\sigma}_1^2}{n_1^2}\sum_{i:\, D_i=1} w_i^2 + \frac{\hat{\sigma}_0^2}{n_0^2}\sum_{i:\, D_i=0} w_i^2\right) \\ &- 2\left(\frac{\hat{\sigma}_1^2}{n_1}\sum_{i:\, D_i=1} w_i + \frac{\hat{\sigma}_0^2}{n_0}\sum_{i:\, D_i=0} w_i\right)\Bigg)\end{aligned}$$

But $\sum_{i:\, D_i=1} w_i = n_1$ and $\sum_{i:\, D_i=0} w_i = n_0$, so term 5 ultimately simplifies to a constant value (as best we can estimate it). It then can be dropped from the optimization problem in the same way term 6 was. We have thus successfully rewritten our minimization of the conditional MSE as

$$\operatorname{argmin}_w \left(\left(|\text{EWC Bias}^{\text{ATE}}|\right)^2 + \left(\frac{\hat{\sigma}_1^2}{n_1^2}\sum_{i:\, D_i=1} w_i^2 + \frac{\hat{\sigma}_0^2}{n_0^2}\sum_{i:\, D_i=0} w_i^2\right)\right)$$

2.2 Properties of the EWC Bias

The EWC bias is not defined as a difference in imbalances only to match the term in our minimization–it has also been defined such that its expectation is equal to the formal statistical bias of our estimator $\hat{\tau}_{\text{wdim}}$, given that we observe X and D. Note that:

$$\begin{aligned}\hat{\tau}_{\text{wdim}}^{\text{ATE}} - \text{SATE} &= \left(\frac{1}{n_1}\sum_{i:\, D_i=1} w_i Y_i(1) - \frac{1}{n_0}\sum_{i:\, D_i=0} w_i Y_i(0)\right) - \left(\frac{1}{n}\sum_{i=1}^{n} Y_i(1) - \frac{1}{n}\sum_{i=1}^{n} Y_i(0)\right) \\ &= \left(\frac{1}{n}\sum_{i=1}^{n} Y_i(0) - \frac{1}{n_0}\sum_{i:\, D_i=0} w_i Y_i(0)\right) - \left(\frac{1}{n}\sum_{i=1}^{n} Y_i(1) - \frac{1}{n_1}\sum_{i:\, D_i=1} w_i Y_i(1)\right) \\ &= \left(\frac{1}{n}\sum_{i=1}^{n} f_0(X_i) - \frac{1}{n_0}\sum_{i:\, D_i=0} w_i f_0(X_i)\right) - \left(\frac{1}{n}\sum_{i=1}^{n} f_1(X_i) - \frac{1}{n_1}\sum_{i:\, D_i=1} w_i f_1(X_i)\right) \\ &+ \left(\frac{1}{n}\sum_{i=1}^{n} \epsilon_i(0) - \frac{1}{n_0}\sum_{i:\, D_i=0} w_i \epsilon_i(0)\right) - \left(\frac{1}{n}\sum_{i=1}^{n} \epsilon_i(1) - \frac{1}{n_1}\sum_{i:\, D_i=1} w_i \epsilon_i(1)\right) \\ &= \text{imbal}_0^{\text{ATE}}(w, f_0(X), D) - \text{imbal}_1^{\text{ATE}}(w, f_1(X), D) \\ &+ \left(\frac{1}{n}\sum_{i=1}^{n} \epsilon_i(0) - \frac{1}{n_0}\sum_{i:\, D_i=0} w_i \epsilon_i(0)\right) - \left(\frac{1}{n}\sum_{i=1}^{n} \epsilon_i(1) - \frac{1}{n_1}\sum_{i:\, D_i=1} w_i \epsilon_i(1)\right)\end{aligned}$$

The bias of an estimator $\hat{\theta}$ of a parameter θ is defined as

$$\text{bias}(\hat{\theta}) = E[\hat{\theta}] - \theta$$

So the bias of our estimator, the weighted difference in means, given we observe our data, is

$$\text{bias}(\hat{\tau}^{\text{ATE}}_{\text{wdim}}) = E[\hat{\tau}^{\text{ATE}}_{\text{wdim}} \mid X, D] - \text{ATE}$$

If we take the the expectation of the previous equation:

$$\begin{aligned} E[\hat{\tau}_{\text{wdim}} - \text{SATE}] &= E\Bigg[\text{imbal}^{\text{ATE}}_0(w, f_0(X), D) - \text{imbal}^{\text{ATE}}_1(w, f_1(X), D) \\ &\quad + \left(\frac{1}{n}\sum_{i=1}^{n} \epsilon_i(0) - \frac{1}{n_0}\sum_{i:\ D_i=0} w_i\epsilon_i(0)\right) \\ &\quad - \left(\frac{1}{n}\sum_{i=1}^{n} \epsilon_i(1) - \frac{1}{n_1}\sum_{i:\ D_i=1} w_i\epsilon_i(1)\right) \mid X, D\Bigg] \\ &= E[\text{imbal}^{\text{ATE}}_0(w, f_0(X), D) - \text{imbal}^{\text{ATE}}_1(w, f_1(X), D) \mid X, D] \end{aligned}$$

Because our error terms are expectation zero. And equivalently:

$$\begin{aligned} E[\hat{\tau}_{\text{wdim}} - \text{SATE}] &= E[\hat{\tau}_{\text{wdim}}] - E[\text{SATE}] && \text{linearity of expectation} \\ &= E[\hat{\tau}_{\text{wdim}}] - \text{ATE} && \text{SATE is unbiased} \\ &= \text{bias}(\hat{\tau}_{\text{wdim}}) && \text{definition of statistical bias} \end{aligned}$$

$$\implies \text{bias}(\hat{\tau}_{\text{wdim}}) = E[\text{imbal}^{\text{ATE}}_0(w, f_0(X), D) - \text{imbal}^{\text{ATE}}_1(w, f_1(X), D)]$$

So we have constructed that

$$\text{EWC Bias} = \text{imbal}^{\text{ATE}}_0(w, f_0(X), D) - \text{imbal}^{\text{ATE}}_1(w, f_1(X), D)$$

as desired.

2.3 Finding the Worst Case EWC Bias

Before continuing with the minimization problem, we will consider a secondary optimization problem, this one a maximization over the EWC bias. Our final minimization problem is going to be done on the worst case EWC bias. Specifically, we are looking for the maximum EWC bias as a function of β_d near $\hat{\beta}_d$ in $\mathbb{R}^{p_d}$. We'll say that β_d is 'near' our estimate $\hat{\beta}_d$ when β_d falls within an ellipsoid centered at $\hat{\beta}_d$ of a certain size with probability q. Because the set is centered around $\hat{\beta}_d$, we can choose to instead express it as new set at the origin, shifted by $\hat{\beta}_d$. We call this new set $S_d(q)$, with the property:

$$\begin{aligned} P(\beta_d \in S_d(q) + \hat{\beta}_d) &= q \\ P(\beta_d - \hat{\beta}_d \in S_d(q)) &= q \end{aligned}$$

We equivalently have that

$$P(\hat{\beta}_d - \beta_d \in S_d(q)) = q$$

because ellipsoids are symmetric and $S_d(q)$ is centered at the origin. We will further construct $S_d(q)$ such that when scaled by $\hat{V}_{\beta_d}^{-\frac{1}{2}}$, it becomes a closed hypersphere with radius $R_d(q)$ centered at the origin, that a standard multivariate normal random variable will fall in with probability q:

$$\begin{aligned} P(\hat{\beta}_d - \beta_d \in S_d(q)) &= P(\hat{V}_{\beta_d}^{-\frac{1}{2}}(\hat{\beta}_d - \beta_d) \in \hat{V}_{\beta_d}^{-\frac{1}{2}} S_d(q)) \\ &\approx P(\mathcal{N}(\vec{0}_{p_d}, I_{p_d}) \in \hat{V}_{\beta_d}^{-\frac{1}{2}} S_d(q)) \\ &\approx q \end{aligned}$$

Note that this approximation holds by the asymptotic normality of $\hat{\beta}_d$. This has been established by the central limit theorem for OLS (Grenander, 1954). In other words, $\hat{V}_{\beta_d}^{-\frac{1}{2}}(\hat{\beta}_d - \beta_d) \xrightarrow{\mathcal{D}} \mathcal{N}(\vec{0}_{p_d}, I_{p_d})$ for an asymptotically normal $\hat{\beta}_d$. However, we cannot say that $\hat{\beta}_d$ for KRLS is asymptotically normal–the length of $\hat{\beta}_d$ for KRLS is n_d, so the limit as $n \to \infty$ in the central limit theorem is not well defined. In other words, we aren't really "approaching" a specific distribution for $\hat{\beta}_d$ when our regression method is KRLS. However, if we assume that our error terms are conditionally normal with expectation zero and finite variance, we have that

$$\begin{aligned} \epsilon_i(d) \mid X &\sim \mathcal{N}(0, \sigma_i(d)^2) \\ \hat{\beta}_d \mid X &= (K_d + \lambda I_{n_d})^{-1} Y \mid X \\ \hat{\beta}_d^{(i)} \mid X_i &= (K_d^{(i)} + \lambda I_{n_d})^{-1} Y_i \mid X_i \\ &= (K_d^{(i)} + \lambda I_{n_d})^{-1} f_d(X_i) + (K_d^{(i)} + \lambda I_{n_d})^{-1} \epsilon(d) \mid X_i \\ &\sim \mathcal{N}((K_d^{(i)} + \lambda I_{n_d})^{-1} f_d(X_i), (K_d^{(i)} + \lambda I_{n_d})^{-\frac{1}{2}} \sigma_i(d)^2) \end{aligned}$$

Where K_d is the positive semi-definite kernel matrix defined in the background section on KRLS and $K_d^{(i)}$ is its ith row[1][2].

And so the desired probabilistic property will still hold. The motivation of TFB may also work for any asymptotically normal $\hat{f}_d(X_i)$, not just an asymptotically normal coefficient vector, due to our generalization of the true functional form of $\hat{f}_d(X)$ through $g_d(X)$ [3]. Even when none of these properties are necessarily the case, though, we are still able to use the probabilistic approximation as motivation for our estimator.

Then $P(\mathcal{N}(\vec{0}_{p_d}, I_{p_d}) \in \hat{V}_{\beta_d}^{-\frac{1}{2}} S_d(q))$ is equivalent to determining the probability that the euclidian distance of the normal random variable from the origin is less than $R_d(q)$, or

$$\begin{aligned} P(\|\mathcal{N}(\vec{0}_{p_d}, I_{p_d})\|_2 \leq R_d(q)) &= P(\|\mathcal{N}(\vec{0}_{p_d}, I_{p_d})\|_2^2 \leq R_d(q)^2) \\ &= q \end{aligned}$$

But then $\|\mathcal{N}(\vec{0}_{p_d}, I_{p_d})\|^2$ is a sum of squares of random normal variables, causing it to follow the $\chi^2_{p_d}$ distribution. So $R_d(q)^2$ is given by the qth quantile, and $R_d(q)$ is given by its root, which we notate as $\sqrt{Q_q(\chi^2_{p_d})}$. Now we have a representation of $S_d(q)$:

$$\begin{aligned} \hat{V}_{\beta_d}^{-\frac{1}{2}} S_d(q) &= \left\{ z_d \in \mathbb{R}^{p_d} \mid \|z_d\|_2 \leq \sqrt{Q_q(\chi^2_{p_d})} \right\} \\ S_d(q) &= \left\{ \hat{V}_{\beta_d}^{\frac{1}{2}} z_d \in \mathbb{R}^{p_d} \mid \|z_d\|_2 \leq \sqrt{Q_q(\chi^2_{p_d})} \right\} \end{aligned}$$

Returning to finding the worst case EWC bias, we see that because it's squared we

[1]K_d is the matrix of similarity between observations with $i\colon D_i = d$, making it a subset of the kernel matrix K of the entire dataset X. For the clarity of notation in the above statement of normality, I have assumed that $1 \leq i < j \leq n$ for all $i\colon D_i = d$ and $j\colon D_j = 1 - d$

[2]Note that K_d is not $g_d(X)$–K_d is necessarily n_d by n_d to solve for $\hat{\beta}_d$ (analogous to $\hat{c}$ in the background section), while $g_d(X)$ is necessarily n by n_d because of our restriction on its shape and because it must give estimates of f_d through the matrix product $g_d(X)\hat{\beta}_d$. A perhaps more intuitive way to think of this distinction is that K_d is the "training set" for $\hat{f}_d$, while $g_d(X)$ is the extension of the domain of $\hat{f}_d$ to the entire dataset. So while K_d only contains measures of similarity of treated units to treated units or of controlled units to controlled units, $g_d(X)$ respectively contains measures of similarity of all units to treated units or of all units to controlled units.

[3]Consider an asymptotically normal $\hat{f}_d(X_i)$. Then the associated vector of predicted outcomes $\hat{f}_d(X)$ has an asymptotically normal multivariate distribution (this may require independent observations). Let's perform TFB on a new characterization of f_d, which I will superscript with a prime symbol. Let $\hat{\beta}'_d = \hat{f}_d(X)$, and let $g'_d(X) = I_n$. Then $\hat{\beta}'_d$ is asymptotically normal, and I suspect that the minimization for TFB is equivalent. Notably, KRLS has been proven to have an asymptotically normal $\hat{f}_d(X_i)$ (Hainmueller & Hazlett, 2014).

can write it as a maximization problem

$$\begin{aligned}
&\max_{(\beta_0,\beta_1)\text{near}(\hat\beta_0,\hat\beta_1)} \left|\text{EWC Bias}^{\text{ATE}}\right| \\
&\max_{(\beta_0,\beta_1)\text{near}(\hat\beta_0,\hat\beta_1)} \left|\text{imbal}_1^{\text{ATE}}(w, f_1(X), D) - \text{imbal}_0^{\text{ATE}}(w, f_0(X), D)\right| \\
\approx &\max_{\|z_d\|_2 \le \sqrt{Q_q(\chi^2_{p_d})}} \left|\text{imbal}_1^{\text{ATE}}(w, f_1(X), D) - \text{imbal}_0^{\text{ATE}}(w, f_0(X), D)\right|
\end{aligned}$$

for standardized p_d by 1 vectors $z_d = \hat V^{-\frac{1}{2}}(\hat\beta_d - \beta_d)$. Now we need to set a reasonable upper bound on the expression. We note that β_d can be brought outside of the imbalance function:

$$\begin{aligned}
\text{imbal}_d^{\text{ATE}}(w, f_d(X), D) &= \frac{1}{n}\sum_{i=1}^n f_d(X_i) - \frac{1}{n_d}\sum_{i:\, D_i=d} w_i f_d(X_i) \\
&= \frac{1}{n}\sum_{i=1}^n g_d(X_i)\beta_d - \frac{1}{n_d}\sum_{i:\, D_i=d} w_i g_d(X_i)\beta_d \\
&= \left(\frac{1}{n}\sum_{i=1}^n g_d(X_i) - \frac{1}{n_d}\sum_{i:\, D_i=d} w_i g_d(X_i)\right)\beta_d \\
&= \left(\frac{1}{n}\sum_{i=1}^n g_d(X_i) - \frac{1}{n_d}\sum_{i:\, D_i=d} w_i g_d(X_i)\right)\beta_d \\
&= \text{imbal}_d^{\text{ATE}}(w, g_d(X), D)\beta_d
\end{aligned}$$

Through some substitution and algebraic manipulation, we have:

$$\begin{aligned}
&\max_{\|z_d\|_2 \le \sqrt{Q_q(\chi^2_{p_d})}} \left| \text{imbal}_1^{\text{ATE}}(w, f_1(X), D) - \text{imbal}_0^{\text{ATE}}(w, f_0(X), D) \right| \\
&= \max_{\|z_d\|_2 \le \sqrt{Q_q(\chi^2_{p_d})}} \left| \text{imbal}_1^{\text{ATE}}(w, g_1(X), D)\beta_1 - \text{imbal}_0^{\text{ATE}}(w, g_0(X), D)\beta_0 \right| \\
&= \max_{\|z_d\|_2 \le \sqrt{Q_q(\chi^2_{p_d})}} \Big| \text{imbal}_1^{\text{ATE}}(w, g_1(X), D)(\beta_1 - \hat{\beta}_1) + \text{imbal}_1^{\text{ATE}}(w, g_1(X), D)\hat{\beta}_1 \\
&\qquad - \Big(\text{imbal}_0^{\text{ATE}}(w, g_0(X), D)(\beta_0 - \hat{\beta}_0) + \text{imbal}_0^{\text{ATE}}(w, g_0(X), D)\hat{\beta}_0\Big) \Big| \\
&= \max_{\|z_d\|_2 \le \sqrt{Q_q(\chi^2_{p_d})}} \Big| - \text{imbal}_1^{\text{ATE}}(w, g_1(X), D)\hat{V}_{\beta_1}^{\frac{1}{2}}\hat{V}_{\beta_1}^{-\frac{1}{2}}(\hat{\beta}_1 - \beta_1) + \text{imbal}_1^{\text{ATE}}(w, g_1(X), D)\hat{\beta}_1 \\
&\qquad + \text{imbal}_0^{\text{ATE}}(w, g_0(X), D)\hat{V}_{\beta_0}^{\frac{1}{2}}\hat{V}_{\beta_0}^{-\frac{1}{2}}(\hat{\beta}_0 - \beta_0) - \text{imbal}_0^{\text{ATE}}(w, g_0(X), D)\hat{\beta}_0 \Big| \\
&= \max_{\|z_d\|_2 \le \sqrt{Q_q(\chi^2_{p_d})}} \Big| \text{imbal}_1^{\text{ATE}}(w, g_1(X), D)\hat{V}_{\beta_1}^{\frac{1}{2}} z_1 - \text{imbal}_1^{\text{ATE}}(w, g_1(X), D)\hat{\beta}_1 \\
&\qquad - \text{imbal}_0^{\text{ATE}}(w, g_0(X), D)\hat{V}_{\beta_0}^{\frac{1}{2}} z_0 + \text{imbal}_0^{\text{ATE}}(w, g_0(X), D)\hat{\beta}_0 \Big|
\end{aligned}$$

Then we set an upper bound by the triangle inequality.

$$\begin{aligned}
&\max_{\|z_d\|_2 \le \sqrt{Q_q(\chi^2_{p_d})}} \Big| \text{imbal}_1^{\text{ATE}}(w, g_1(X), D)\hat{V}_{\beta_1}^{\frac{1}{2}} z_1 - \text{imbal}_1^{\text{ATE}}(w, g_1(X), D)\hat{\beta}_1 \\
&\qquad - \text{imbal}_0^{\text{ATE}}(w, g_0(X), D)\hat{V}_{\beta_0}^{\frac{1}{2}} z_0 + \text{imbal}_0^{\text{ATE}}(w, g_0(X), D)\hat{\beta}_0 \Big| \\
&\le \max_{\|z_d\|_2 \le \sqrt{Q_q(\chi^2_{p_d})}} \Big[\left| \text{imbal}_1^{\text{ATE}}(w, g_1(X), D)\hat{V}_{\beta_1}^{\frac{1}{2}} z_1 \right| + \left| \text{imbal}_1^{\text{ATE}}(w, g_1(X), D)\hat{\beta}_1 \right| \\
&\qquad + \left| \text{imbal}_0^{\text{ATE}}(w, g_0(X), D)\hat{V}_{\beta_0}^{\frac{1}{2}} z_0 \right| + \left| \text{imbal}_0^{\text{ATE}}(w, g_0(X), D)\hat{\beta}_0 \right| \Big]
\end{aligned}$$

We know by the Cauchy-Schwarz inequality that for vectors u, v, $|u^T v| \le \|u\|_2 \|v\|_2$. Specifically, we know that this becomes an equality when $u \parallel v$. And we know that we can find a z_d parallel to $\text{imbal}_d^{\text{ATE}}(w, g_d(X), D)\hat{V}_{\beta_d}^{\frac{1}{2}}$ within the domain $\hat{V}_{\beta_d}^{-\frac{1}{2}} S_d(q)$ of z_d, because that domain is a hypersphere. And finally, note that

$\text{imbal}_d^{\text{ATE}}(w, g_d(X), D)\hat{V}_{\beta_d}^{\frac{1}{2}}$ is a constant in our maximization, so

$$\begin{aligned}
&\max_{\|z_d\|_2 \le \sqrt{Q_q(\chi^2_{p_d})}} \left|\text{imbal}_d^{\text{ATE}}(w, g_d(X), D)\hat{V}_{\beta_d}^{\frac{1}{2}} z_d\right| \\
&= \max_{\|z_d\|_2 \le \sqrt{Q_q(\chi^2_{p_d})}} \|\text{imbal}_d^{\text{ATE}}(w, g_d(X), D)\hat{V}_{\beta_d}^{\frac{1}{2}}\|_2 \|z_d\|_2 \\
&\qquad =\|\text{imbal}_d^{\text{ATE}}(w, g_d(X), D)\hat{V}_{\beta_d}^{\frac{1}{2}}\|_2 \max_{\|z_d\|_2 \le \sqrt{Q_q(\chi^2_{p_d})}} \|z_d\|_2 \\
&\qquad =\|\text{imbal}_d^{\text{ATE}}(w, g_d(X), D)\hat{V}_{\beta_d}^{\frac{1}{2}}\|_2 \sqrt{Q_q(\chi^2_{p_d})}
\end{aligned}$$

This gives us the solution to our maximization problem.

$$\begin{aligned}
\max_{\|z_d\|_2 \le \sqrt{Q_q(\chi^2_{p_d})}} &\left|\text{imbal}_1^{\text{ATE}}(w, g_1(X), D)\hat{V}_{\beta_1}^{\frac{1}{2}} z_1\right| + \left|\text{imbal}_1^{\text{ATE}}(w, g_1(X), D)\hat{\beta}_1\right| \\
&+ \left|\text{imbal}_0^{\text{ATE}}(w, g_0(X), D)\hat{V}_{\beta_0}^{\frac{1}{2}} z_0\right| + \left|\text{imbal}_0^{\text{ATE}}(w, g_0(X), D)\hat{\beta}_0\right| \\
&= \left\|\text{imbal}_1^{\text{ATE}}(w, g_1(X), D)\hat{V}_{\beta_1}^{\frac{1}{2}}\right\|_2 \sqrt{Q_q(\chi^2_{p_1})} + \left|\text{imbal}_1^{\text{ATE}}(w, g_1(X), D)\hat{\beta}_1\right| \\
&+ \left\|\text{imbal}_0^{\text{ATE}}(w, g_0(X), D)\hat{V}_{\beta_0}^{\frac{1}{2}}\right\|_2 \sqrt{Q_q(\chi^2_{p_0})} + \left|\text{imbal}_0^{\text{ATE}}(w, g_0(X), D)\hat{\beta}_0\right|
\end{aligned}$$

2.4 Final Statement of the Minimization for the ATE

We can now bring back the square and error terms and minimize the resulting expression in order to find optimal weights! Our final minimization problem for the ATE is therefore:

$$\begin{aligned}\operatorname{argmin}_w \Bigg[\Bigg(& \sqrt{Q_q(\chi^2_{p_1})} \cdot \left\| \text{imbal}_1^{\text{ATE}}(w, g_1(X), D) \hat{V}_{\beta_1}^{\frac{1}{2}} \right\|_2 \\ & + \left| \text{imbal}_1^{\text{ATE}}(w, g_1(X), D) \hat{\beta}_1 \right| \\ & + \sqrt{Q_q(\chi^2_{p_0})} \cdot \left\| \text{imbal}_0^{\text{ATE}}(w, g_0(X), D) \hat{V}_{\beta_0}^{\frac{1}{2}} \right\|_2 \\ & + \left| \text{imbal}_0^{\text{ATE}}(w, g_0(X), D) \hat{\beta}_0 \right| \Bigg)^2 \\ & + \frac{\hat{\sigma}_0^2}{n_0^2} \sum_{i:\, D_i=0} w_i^2 + \frac{\hat{\sigma}_1^2}{n_1^2} \sum_{i:\, D_i=1} w_i^2 \Bigg]\end{aligned}$$

Note that this implies we will be choosing weights that aim for 4 things:

1. Balance between the covariates $g_d(X)$ (notably, if the covariates of $g_d(X)$ are balanced, the following two more complex targets are balanced as well). Suppose the average of the jth covariate in the entire dataset is particularly different from its unweighted average in the respective treatment or control group due to its distribution being skewed relative to the entire dataset's distribution. Then there are certain observations overrepresenting a certain range of covariate values, which we can weight down or weight up in order to make the weighted distribution more similar to the entire dataset's distribution in shape and therefore in mean. We would then expect our optimization problem to give us weights that do this.

2. Balance between the predicted outcomes $\hat{f}_d(X)$. If the distribution of $\hat{f}_d(X)$ in the treatment or control group is skewed relative to the entire dataset, we expect TFB to weight its observations to make the distributions more similar.

3. Balance between the covariates scaled by the root of their covariance $\hat{V}_{\beta_d}^{\frac{1}{2}} g_d(X)$. If the variance/covariance of one of the initial regression's coefficients is particularly large, we expect TFB to choose weights that prioritize balance in the associated dimensions of $g_d(X)$.

4. Low variance in the weights. Our error term tries to minimize the sum of the squared weights, so we would expect our optimization problem to give us weights as close to one (the expected value of a given w_i) as possible.

To see how we rewrite this convex optimization problem as a linear objective function with constraints for the purpose of solving it with `Rmosek`, see Appendix A.

Chapter 3

Simulation Study

To demonstrate that the `tfb` package can perform unbiased estimation of the ATT, ATC, and ATE, we will perfom two simulation studies and compare the causal estimates retrieved by targeted function balancing to the sample estimators of the SATT, SATC, and SATE, which are unbiased for their respective estimands with ideal efficiency. We will also compare TFB's estimates to the naive DIM and the estimates of kernel balancing (KBAL), another method of approximate balancing weights (Hazlett, 2020). The first data generating process (DGP) will have potential outcome CEFs that are linear in the covariates, while the second DGP will have potential outcome CEFs that are nonlinear in the covariates. Both will have treatment assignment mechanisms that are *not* independent of the covariates, which therefore makes the observed outcome a function of both the treatment and the covariates. Note that neither of the DGPs violate conditional ignorability, so all the assumptions for unbiased estimation are met.

In order to run our code, we'll use the `tfb` package, and `tidyverse` for visualization and data processing (Wickham et al., 2019). The full code is provided in the code appendix and can be used to reproduce the results. Each DGP has been iterated 400 times in order to produce the spread of possible values shown.

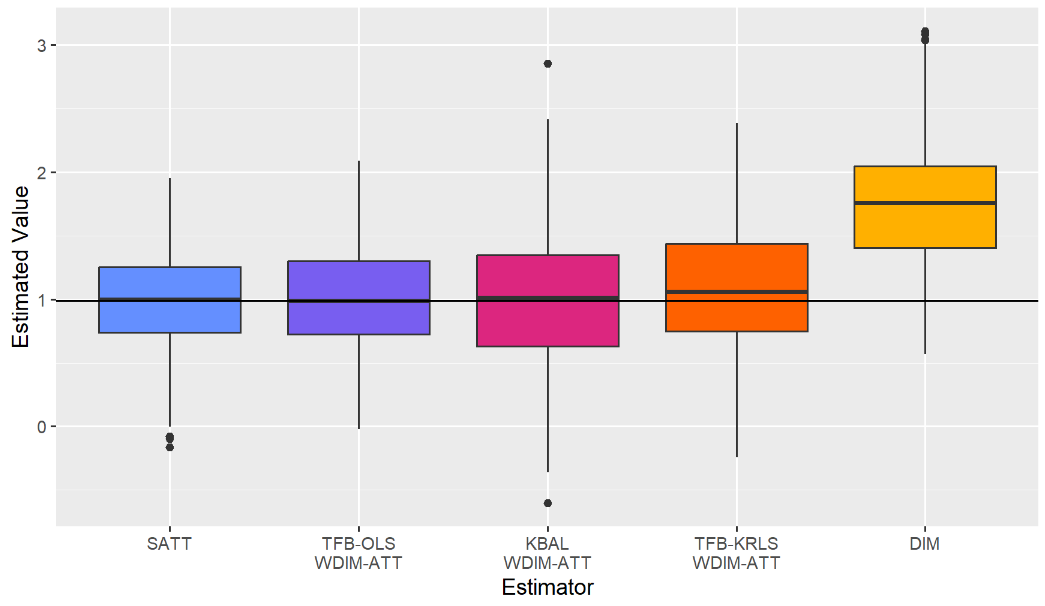
3.1 DGP 1

The first DGP is simpler, with f_ds that are a linear combination of the covariates. It's defined as follows:

$$\begin{aligned}
n &= 200 \\
X_{i,1} &\sim \mathcal{N}(0,1) \\
X_{i,2} &\sim \mathcal{N}(0,0.0625) \\
\epsilon_0(X_i) &\sim \mathcal{N}(0,4) \\
\epsilon_1(X_i) &\sim \mathcal{N}(0,2.25) \\
Y_i(0) &= X_{i,1} + 2\cdot X_{i,2} + \epsilon_0(X_i) \\
Y_i(1) &= X_{i,1} + 2\cdot X_{i,2} + 3\cdot X_{i,1} + X_{i,2} + \epsilon_1(X_i) \\
P(D_i=1) &= \frac{e^{0.7\cdot X_{i,1}+X_{i,2}}}{1+e^{0.7\cdot X_{i,1}+X_{i,2}}} \\
P(D_i=0) &= 1 - P(D_i=1) \\
Y_i &= Y_i(1)\cdot D_i + Y_i(0)\cdot(1-D_i)
\end{aligned}$$

We expect TFB with OLS as an initial regression (TFB-OLS) to perform very well on it, as it correctly specifies the CEF by constructing a model for f_d that is linear in the covariates. We comparatively expect TFB with KRLS as an initial regression (TFB-KRLS) to perform worse, as it does not have nearly enough observations to estimate the CEFs well, and it lacks the parametric assumption of linearity. We would expect good performance from KBAL, although it should generally have a higher variance than TFB. The baseline for performance will be given by the sample ATT, ATC, and ATE as the theoretically most efficient estimators of their respective estimands. Estimators are ordered from left to right by the magnitude of their bias. The sample estimators are unbiased, so we will use their means as the "true" values of the ATT, ATE, and ATC. So for a sample estimator $\hat{\theta}$ of a given estimand θ, and an arbitrary estimator $\hat{\tau}$ of θ, the further right an estimator's boxplot is in the figure, the greater its respective $\left|E[\hat{\tau}] - E[\hat{\theta}]\right|$. Starting with the ATT:

Figure 3.1.1: Comparison of ATT Estimator Performance on a Linear DGP

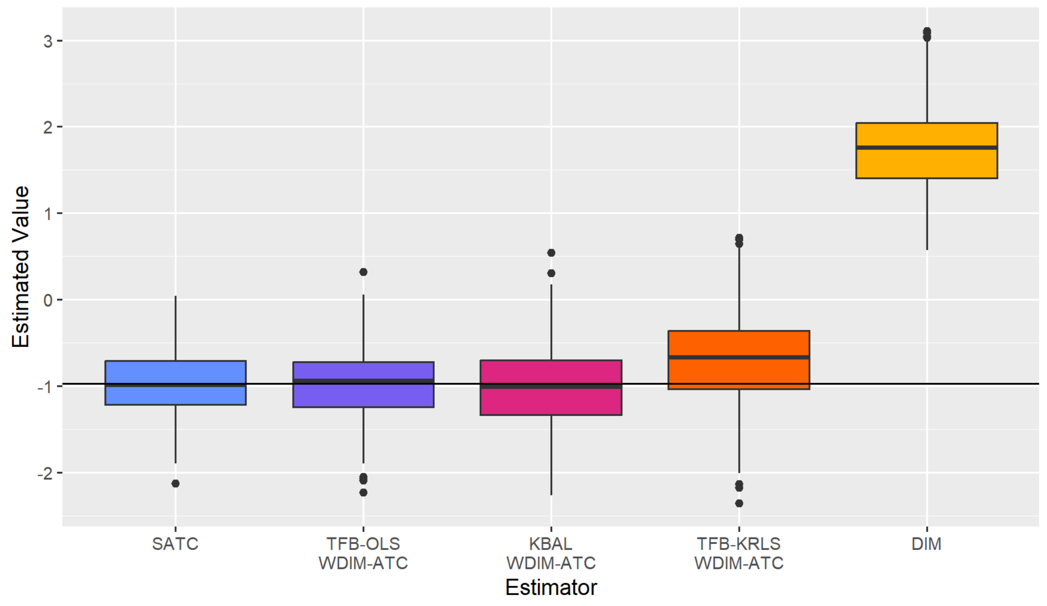


Random data with linear CEFs was simulated for a sample size of $n = 200$ across 400 iterations, and the distributions of estimates for the ATT across 5 different estimators are shown as boxplots, with the mean value of the SATT serving as the benchmark.

We see, as expected, that TFB-OLS has the best performance in both bias and variance, because its strong parametric assumption of linearity is met by the CEFs of f_d. TFB-KRLS is slightly more biased than KBAL, though KBAL has higher variance. Regardless, both of the kernel methods perform at this low sample size.

Moving on to the ATC, we see similar results.

Figure 3.1.2: Comparison of ATC Estimator Performance on a Linear DGP

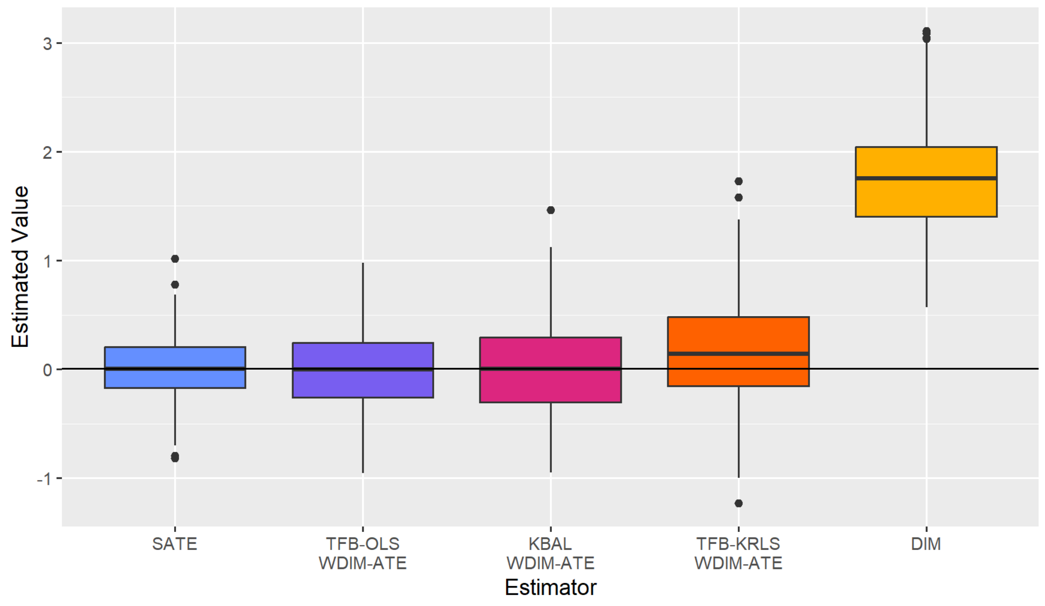


Random data with linear CEFs was simulated for a sample size of $n = 200$ across 400 iterations, and the distributions of estimates for the ATC across 5 different estimators are shown as boxplots, with the mean value of the SATC serving as the benchmark.

TFB-OLS is the closest approximation of the SATC, and KBAL and TFB-KRLS trails somewhat close behind. The only truly poor estimator is the naive DIM.

As for the ATE:

Figure 3.1.3: Comparison of ATE Estimator Performance on a Linear DGP



Random data with linear CEFs was simulated for a sample size of $n = 200$ across 400 iterations, and the distributions of estimates for the ATE across 5 different estimators are shown as boxplots, with the mean value of the SATE serving as the benchmark.

We see that the same pattern continues to hold, with the correct specification of the functional form making TFB-OLS the best estimator other than the sample estimator. However, this first DGP is comparably easy to balance.

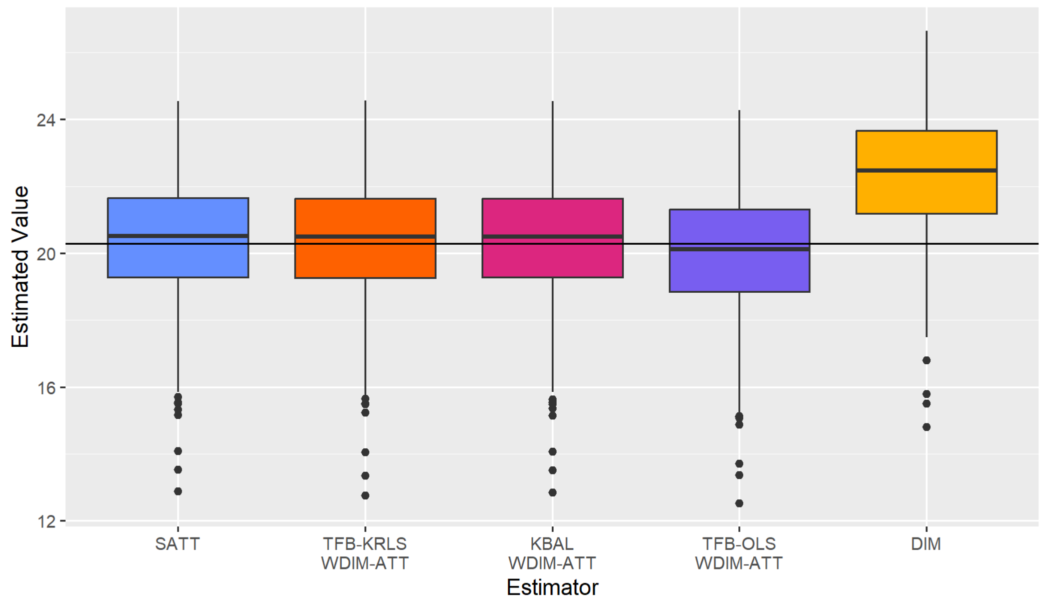
3.2 DGP 2

The second DGP is more complex, and the f_ds have nonlinear polynomial CEFs (specifically, they are saddle surfaces). It's defined as follows:

$$\begin{aligned}
n &= 1000 \\
X_{i,1} &\sim \chi^2_3 - 3 \\
X_{i,2} &\sim \beta(0.5, 0.5) \\
\epsilon_0(X_i) &\sim \mathcal{N}(0, 0.0625) \\
\epsilon_1(X_i) &\sim \mathcal{N}(0, 0.25) \\
Y_i(0) &= (X_{i,1} - 3)(X_{i,2} - 2) + \epsilon_0(X_i) \\
Y_i(1) &= (X_{i,1} - 3)(X_{i,1} + 3)(X_{i,2} - 2)(X_{i,2} + 3) + \epsilon_1(X_i) \\
P(D_i = 1) &= \frac{1}{1 + e^{X_{i,1} \cdot X_{i,2} + 0.5}} \\
P(D_i = 0) &= 1 - P(D_i = 1) \\
Y_i &= Y_i(1) \cdot D_i + Y_i(0) \cdot (1 - D_i)
\end{aligned}$$

We expect TFB-KRLS to perform fairly well on this DGP, as the true functional form is nonlinear but smooth, and there are lots of observations to work with. We also expect KBAL to perform similarly as before, but we expect OLS to become biased due to the misspecification of the functional form.

Figure 3.2.1: Comparison of ATT Estimator Performance on a Nonlinear DGP

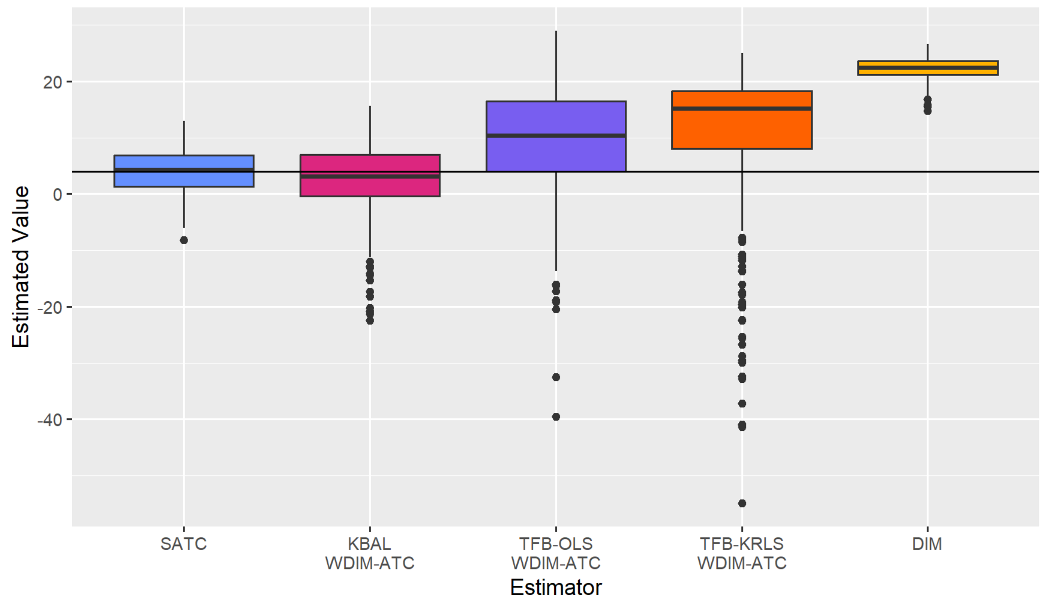


Random data with nonlinear CEFs was simulated for a sample size of $n = 1000$ across 400 iterations, and the distributions of estimates for the ATT across 5 different estimators are shown as boxplots, with the mean value of the SATT serving as the benchmark.

As we hoped, TFB-KRLS performs very well in this scenario, its distribution coming very close to that of the SATT. KBAL is comparable. TFB-OLS performs surprisingly well, and the DIM is far behind as usual. Interestingly, the distribution of the SATT itself is skewed left for this DGP, and none of the estimators have a median matching the SATT's mean, though they are close (with the exception of the DIM).

Moving on to the ATC:

Figure 3.2.2: Comparison of ATC Estimator Performance on a Nonlinear DGP

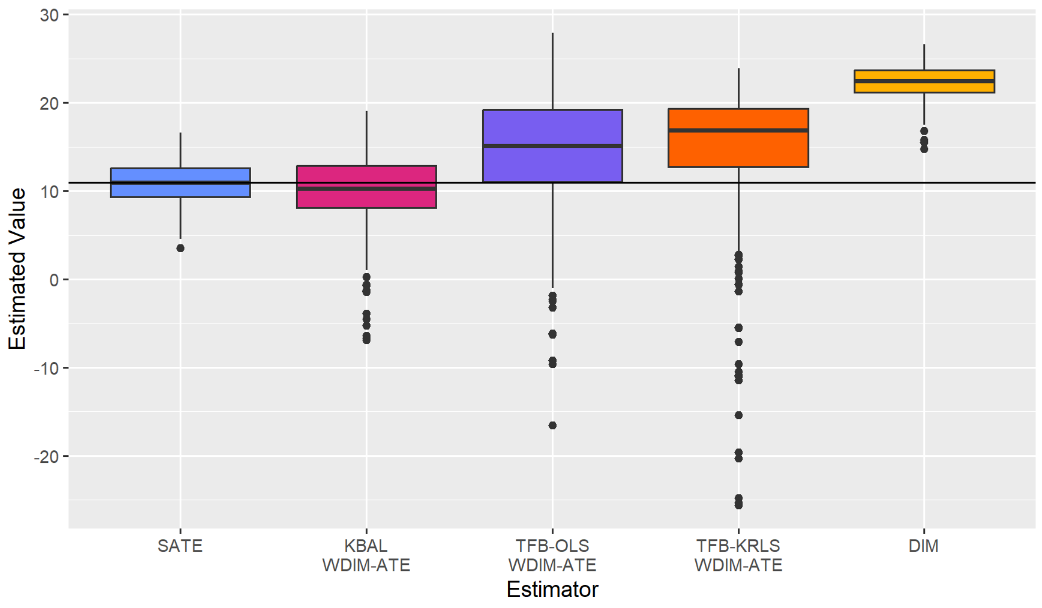


Random data with nonlinear CEFs was simulated for a sample size of $n = 1000$ across 400 iterations, and the distributions of estimates for the ATC across 5 different estimators are shown as boxplots, with the mean value of the SATC serving as the benchmark.

Surprisingly, the expected trend changes when we attempt to estimate the ATC. The SATC is slightly skewed, like the SATT, but the other estimators are no longer comparable. KBAL isn't particularly biased, but it does have a larger variance relative to the SATC. The TFB estimates for this estimand are all highly biased. Notably, TFB-OLS easily outperforms TFB-KRLS in this setting! Considering TFB-OLS can be sensitive to misspecification and the CEFs in this DGP are smooth with a large sample size, it seems like KRLS should not be falling behind. I suspect that this is mostly the fault of the particular DGP being especially difficult to perform estimation with. OLS is a parametric model, so it theoretically increases its bias in settings where the CEF has been misspecified in exchanged for lower variance. Correspondingly, KRLS is flexible, and we would expect it to be able to decrease its bias in exchange for higher variability. We do see increased variability here–KRLS has an absurd number of outliers in its distribution. It seems likely that the regression setting is challenging enough for OLS's increased bias to not matter as much. It's also possible that the higher sample size has worked against KRLS here, and it has overfit on the highly nonlinear trends in the data.

Because estimation of the ATC proved difficult, we expect to see a similar trend with the ATE.

Figure 3.2.3: Comparison of ATE Estimator Performance on a Nonlinear DGP

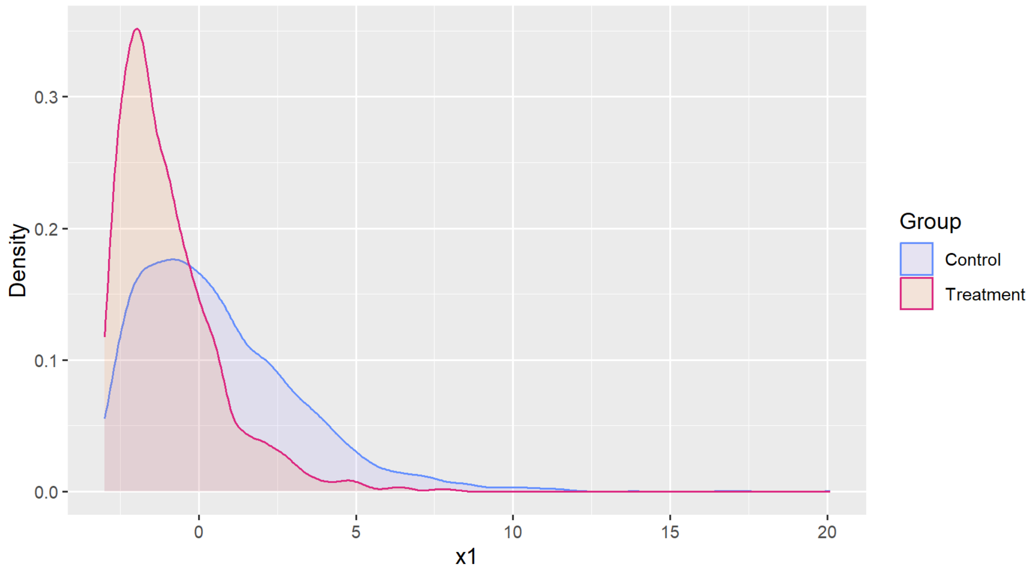


Random data with nonlinear CEFs was simulated for a sample size of $n = 1000$ across 400 iterations, and the distributions of estimates for the ATE across 5 different estimators are shown as boxplots, with the mean value of the SATE serving as the benchmark.

As expected, estimating the ATE for DGP 2 results in an almost identical pattern in estimator behavior compared to estimating the ATC. We again see KBAL doing relatively well compared to TFB and the DIM, TFB-OLS performing relatively well compared to TFB-KRLS, and the naive DIM getting nowhere close.

What actually went wrong with this simulation? That the ATT remained reasonable to estimate gives us a pretty good idea. The issue must lie with the respective treatment and control distributions of covariates. There are plenty of observations in each group, and inspecting the second covariate reveals no issues, but a density plot of the first covariate shows a clear problem.

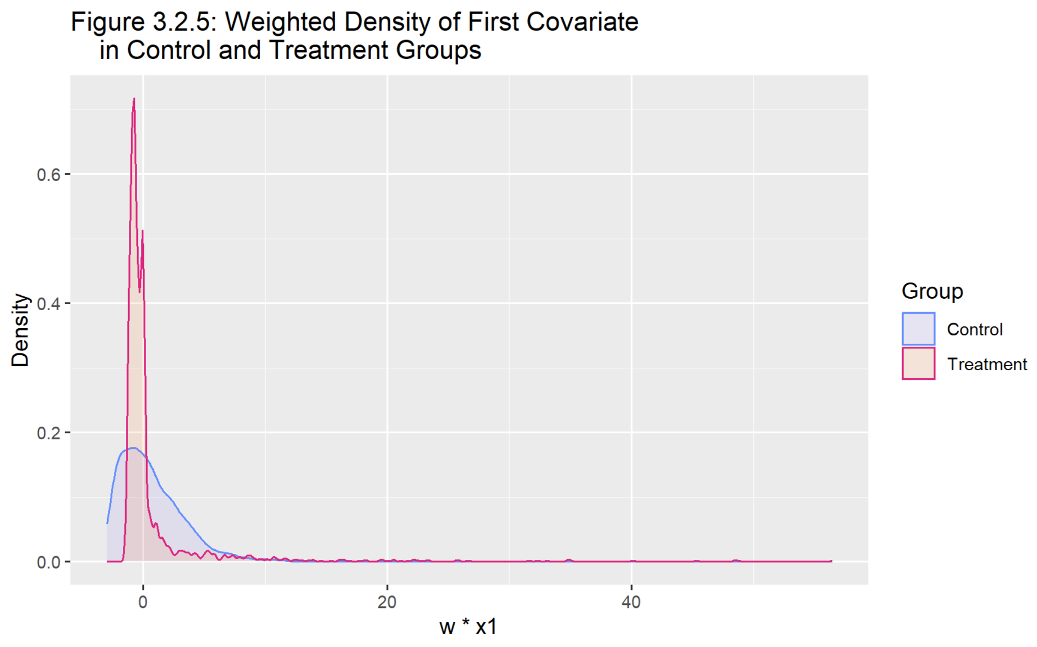
Figure 3.2.4: Density of First Covariate in Control and Treatment Groups



Random data with nonlinear CEFs was simulated for a sample size of $n = 5000$. The distributions of the first covariate in the DGP in the treatment and control groups are shown.

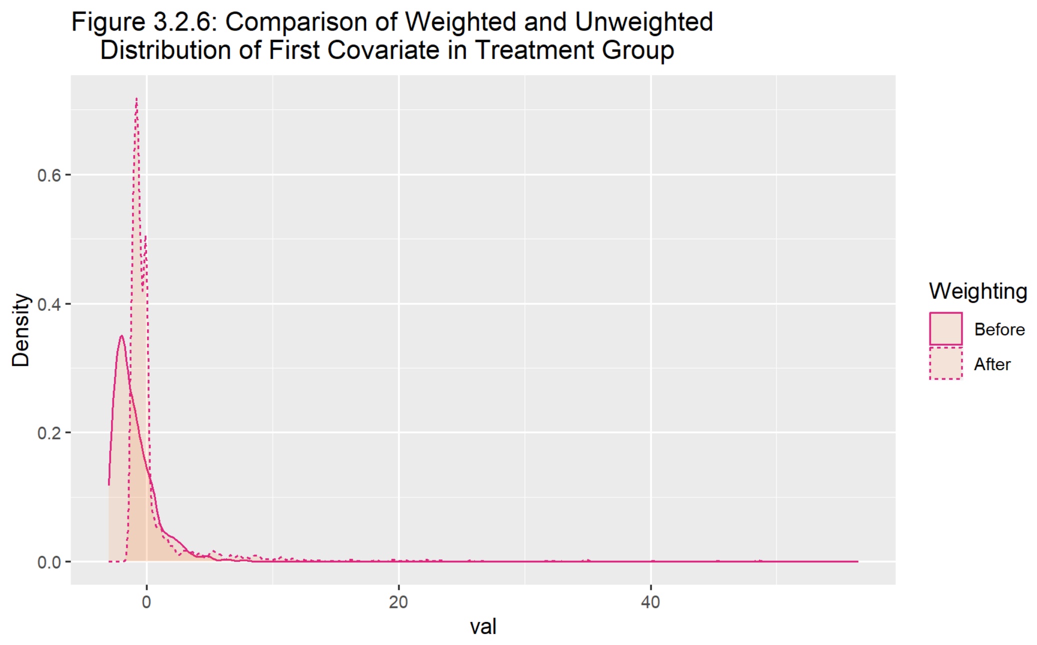
The two distributions are similar in shape, but the frequency of observations being placed in the treatment group drops steeply as `x1` increases, especially relative to the long tail of frequency in the control group. However, both distributions have the same end to their left tail. This means that while the full range of `x1` treatment values are present in the distribution of `x1` control values, the same cannot be said of the converse. By extension, the ATE is also difficult to estimate, because the full range of `x1` values in the data is not represented in the treatment distribution.

An intuitive way to think about this is that while weighted distributions will change in shape and mean, weighting methods will not typically increase a distribution's spread. We can verify this with TFB's best attempt at weighting the treatment distribution, by TFB-OLS:



Random data with nonlinear CEFs was simulated for a sample size of $n = 5000$, weighted by TFB for the ATC with an initial regression choice of OLS. The distribution of the first covariate in the DGP in the weighted treatment and the unweighted control groups are shown.

From this plot, we can see that after weighting for the ATC, the distribution of `x1` in the treatment group approximately matches the mean of `x1` in the control group. However, the right tail, with a range of values that the control group uniquely possesses, has not significantly changed. In a sense, the weights have successfully accounted for the overrepresentation of `x1` values in the treatment group, but not for their underrepresentation. Looking uniquely at the treatment distribution before and after weighting:



Random data with nonlinear CEFs was simulated for a sample size of $n = 5000$, weighted by TFB for the ATC with an initial regression choice of OLS. The distribution of the first covariate in the DGP in the treatment group before and after weighting is shown.

While we do have some sparse observations being weighted up into the right tail of the distribution, the vast majority of the treatment distribution's change in shape lies in it flattening its left tail in order to better match the mean of the control distribution. Every estimator is going to have situations where it excels, and in our second DGP it seems that TFB falls behind other methods like KBAL specifically due to this mismatch in covariate distributions. This is good for the potential researcher to look out for, considering the outsized impact it can have on the consistency of our estimation.

Chapter 4

Package Demonstration

4.1 Package Installation

Part of the work involved for this thesis was coding the TFB optimization problem as an R package `tfb`. The package was built using `roxygen2` (Wickham, Danenberg, Csárdi, & Eugster, 2023). The solution to TFB's optimization problem was coded using `Rmosek` (MOSEK ApS, 2019). To install `tfb`, start by making sure you have the requisite packages installed. This can mostly be done with the following code, though `Rmosek` requires a license for installation.

```r
for (package in c("devtools","roxygen2","KRLS","SparseM")) {
  if (require(package) != T){
    install.packages(package)
    library(package)
  }
}

# Rmosek's installation is more complicated than the above packages.
# However, it is available for free with an educational license.
# See https://docs.mosek.com/latest/rmosek/install-interface.html
require(Rmosek)
```

Once `tfb` is complete potential users will be able to download it from a publically available site. Place the downloaded folder in your current working directory. You can check your current working directory with `getwd()`. You can then install and load it with

```r
install("tfb")
library(tfb)
```

4.2 A Minimum Working Example for TFB

`tfb` is composed of one main function, `tfb()`, and several other scripts that make up different portions of its functionality. For a description of the lower level functionality of `tfb`, see appendix B. `tfb()` has the following parameters:

`X`, the covariate matrix X. This is part of your data and has no default.

`d`, the treatment vector D. This is part of your data and has no default.

`y`, the observed response vector Y. This is part of your data and has no default.

`fit`, the type of initial regression to fit on the data. Currently, this only supports OLS (`"lm"`) and KRLS (`"krls"`). This represents what the user thinks the true functional form of the relationship between Y and X might be, and has no default.

`estimand`, The estimand that the user wishes to estimate, one of the three possible ATT (`"att"`), ATC (`"atc"`), and ATE (`"ate"`). This has no default.

`q`, the probability threshold q. This should be a number in the open interval $(0, 1)$, and it corresponds to how confident the user wishes to be that the true β_d is near the $\hat{\beta}_d$ estimated by the initial regression. The default is the recommended probability threshold of `0.95`.

`rtol`, the `Rmosek` tolerance for optimization, helping to decide when to stop the process. The default is `1e-16`.

`verb`, the verbosity of the `Rmosek` output. Higher integer values print more detail. The default is `0`, for no extra text.

`params`, a list of optional parameters for `tfb()` to give the initial regression. For example, if the user wanted to specify a value of λ for KRLS, they could do that here. The default is `NULL`.

`X_c`, a manually set $g_0(X)$. Make sure its number of rows is equal to the length of D. The default is `NULL`.

`X_t`, a manually set $g_1(X)$. Make sure its number of rows is equal to the length of D. The default is `NULL`.

To see how a user might use TFB, let's use the first DGP from the simulation study. This is given by

```r
makedf <- function(n = 200) {
  x1 <- rnorm(n)              # E 0, V 1
  x2 <- rnorm(n, sd = 0.25) # E 0, V 0.0625
  p <- exp(0.7*x1+x2)/(1+exp(0.7*x1+x2))
  d <- runif(n) < p
  e0 <- rnorm(n, 0, 2)
  e1 <- rnorm(n, 0, 1.5)
  y0 <- x1 + 2 * x2 + e0
  y1 <- x1 + 2 * x2 + 3 * x1 + x2 + e1
  y <- y0 * (1 - d) + y1 * d

  return(cbind(x1,x2,d,y))
} # we can't observe y0 and y1 in practice
```

Note that this DGP satisfies conditional ignorability and we can therefore theoretically retrieve an unbiased causal estimate. So a sample dataframe that we could use for causal inference is given by

```r
set.seed(121)          # setting a random seed so results are replicable
df <- makedf()         # using our dgp to create a matrix of data
df %>%                 # first 6 observations in our data
  data.frame() %>%
  head()
```

Table 4.2.1

i	x1	x2	d	y
1	-0.25536047	-0.1933680	0	1.9297844
2	0.10837472	0.2647281	1	2.6141384
3	0.12778056	-0.1172507	0	-5.3207058
4	-0.08107345	-0.1884090	0	0.2867286
5	-0.71368372	-0.3460796	0	-0.8159717
6	1.61533208	0.1393548	1	5.5531529

This gives us everything we need to find weights for TFB's estimator. Let's say we'd like to estimate the ATE, and that we believe the true functional form to be linear. We can then run

```r
X <- df[,1:2]
d <- df[,3]
y <- df[,4]

set.seed(121)
out <- tfb(
  X = X,            # the covariate matrix is given by
                    # the first two columns of our data
  d = d,            # the treatment vector is given by
                    # the third column of our data
  y = y,            # the observed response vector is given by
                    # the fourth column of our data
  fit = "lm",       # our specification of the initial regression
  estimand = "ate"  # the estimand we want to estimate
)
```

`tfb()` then gives a list as output, composed of:

`$indices`, two lists of indices, each denoting how X was split into two samples for estimation.

`$weights`, two lists of weights, weighting the observations in the first split and the second respectively.

`$beta`, four lists of coefficients, giving the respective forms of $\hat{\beta}_0$ in split 1, $\hat{\beta}_0$ in split 2, $\hat{\beta}_1$ in split 1, and $\hat{\beta}_1$ in split 2. If the ATT or ATC is being estimated, the two $\hat{\beta}_1$ or $\hat{\beta}_0$ vectors won't be given, respectively.

`$covariance`, four estimated covariance matrices of $\hat{\beta}_d$ (two for the ATT or ATC), in the same order as above.

`$error`, four residual vectors of the observed outcomes minus the predicted outcomes (two for the ATT or ATC), in the same order as above.

`$standard_error`, four estimates $\hat{\sigma}_d^2$ of standard error (two for the ATT or ATC), in the same order as above.

`$dim`, three unweighted differences in means $\hat{\tau}_{\text{dim}}$, corresponding to split 1, split 2, and the entire dataset.

`$estimate`, three weighted differences in means $\hat{\tau}_{\text{wdim}}$, corresponding to split 1, split 2, and their mean (the final estimate given by TFB).

`$estimate_variance`, three estimates of the variance of the TFB estimates, in the same order as above.

`$ci`, the lower and upper bound on a $q \cdot 100$ percent confidence interval for the final estimate.

`$p_val`, the p-value of the final estimate under a null hypothesis that there is no average treatment effect.

So in our minimum working example, let's look at the weights of the same observations we sampled above.

```r
# making vector of weights
weight <- rep(0,200)
weight[out$indices$split_1] <- out$weights$weights_1
weight[out$indices$split_2] <- out$weights$weights_2

# adding them to our data
cbind(weight,df) %>%
  data.frame() %>%
  head()
```

Table 4.2.2

i	weight	x1	x2	d	y
1	0.8884137	-0.25536047	-0.1933680	0	1.9297844
2	0.7455230	0.10837472	0.2647281	1	2.6141384
3	1.0820791	0.12778056	-0.1172507	0	-5.3207058
4	0.9360968	-0.08107345	-0.1884090	0	0.2867286
5	0.6921248	-0.71368372	-0.3460796	0	-0.8159717
6	0.4604304	1.61533208	0.1393548	1	5.5531529

So we can see that relative to the average observation in our data, the 3rd observation had its values weighted up, while the 6th observation had its values weighted down, for example.

This is cheating, but we know the true ATE is zero (because the treatment effect is `3 * x1 + x2` and `x1,x2` are mean zero). Comparing the naive estimator $\hat{\tau}_{\text{dim}}$, and our estimator $\hat{\tau}_{\text{wdim},}$, we see that

```
dim <- out$dim$dim
wdim <- out$estimate$est
tibble(
  stat = c("value","magnitude of bias"),
  dim = c(dim,abs(dim)),
  wdim = c(wdim,abs(wdim))
)
```

Table 4.2.3

stat	dim	wdim
value	1.61	-0.401
magnitude of bias	1.61	-0.401

So while our estimate in this case retains some bias due to random noise, it does outperform the naive estimator. And we can further see that our 95 percent confidence interval contains the true value of the ATE

```
tibble(lower = out$ci$lower, upper = out$ci$upper)
```

Table 4.2.4

ci_lower	ci_upper
-1.06	0.263

While also telling us that our data does not suggest an ATE significantly different from zero, under a standard significance threshold of 0.05.

```
tibble(p_val = out$p_val[[1]])
```

Table 4.2.5

p_val
0.2369906

Conclusion

Summary

Targeted Function Balancing (TFB) is a causal inference method for recovering estimates of the causal effect of dichotomous treatments in observational studies, where the covariates confound the effect of the treatment on the outcome (Wainstein, 2022). When conditional ignorability is met, researchers may be able to recover unbiased causal estimates using TFB: specifically, estimates of the average treatment effect on the treated (ATT), the average treatment effect on the controlled (ATC), and the average treatment effect (ATE). TFB does this by modeling the relationship between the outcome and the covariates with an initial regression, and then finding approximate balancing weights. These weights are motivated by the mean squared error between TFB's estimator and the ideal sample estimator of the estimand, and an approximation is minimized with several interpretable targets. TFB tries to minimize the variance of its weights, as well as the worst-case EWC bias, including imbalance in the data, imbalance in the initial regression, and imbalance in covariates with predicted regression coefficients that have high covariance. These properties mean that TFB turns a researcher's question of how best to weight their data into the question of how best to model it. When conditional ignorability is met, a correctly specified relationship between the outcome and the covariates makes TFB effectively unbiased for the estimand of interest. Appropriate specification can also make TFB useful in high-dimensional social science contexts, or other settings where estimation might otherwise be difficult.

TFB has already been developed for the ATT and ATC, but this thesis expands it to the ATE and creates an R package `tfb` for its implementation through code, for the two initial regressions of ordinary least squares (OLS) and kernel regularized least squares (KRLS) (Hainmueller & Hazlett, 2014). In order to help readers recognize settings where TFB can be used to recover causal estimates and provide R code to do so, the thesis covers necessary background for TFB and KRLS, derives TFB's solution to its motivating optimization problem for the ATE, demonstrates its usage on simulated data, and describes how to use the `tfb` package.

Discussion and Further Work

TFB does have some clear limitations on its implementation. To start, the perfect dataset for its usage is hard to come by. Without engineering a specific study design, data that meets the important assumption of conditional ignorability is sparse. This is mostly because conditional ignorability is an idealistic assumption. Conditional ignorability requires there to be no unobserved confounding, but outside of well-understood systems this is not reasonably the case. There will always be some confounder that the researcher cannot measure or reasonably approximate, or cannot measure within their means, or doesn't think to measure, or consciously excludes from the study (for example, data titled as being on "select causes"). Of course, statistical methods are always flawed in practice, so the researcher should do their best to ensure data meets conditional ignorability and interpret their results cautiously.

Non-simulated data that can be used for inference is also hard to come by due to the relative niche-ness of the field. Estimating causal effects accurately is useful and desirable, but people not already interested in statistics are unlikely to know about causal inference methods.

The perfect data for TFB satisfies more than just conditional ignorability. There should also be appropriate distributional overlap in the covariates. This means that when estimating the ATT, for each covariate, we ideally want the full range of values in the treatment group to be represented in the control group. Conversely, for the ATC, we ideally want the full range of covariate values in the control group to be represented in the treatment group. And for the ATE, we want the full range of covariate values in the data to be represented in both the control and the treatment group. Unfortunately, as the number of covariates increases, this naturally becomes more and more difficult to satisfy.

Even beyond the assumptions presented by TFB, each initial regression will also impose its own set of conditions for the data to meet. This is especially true when using initial regressions with strong parametric assumptions such as OLS–when the data isn't truly linear, TFB-OLS is easily biased. More flexible methods like KRLS present their own challenges. For example, when there are insufficient observations, TFB-KRLS's estimator may become subject to bias and subsequent undercoverage (Wainstein, 2022). What sample size is necessary will depend on the noisiness of the data, but TFB-KRLS generally demands a higher n than TFB-OLS.

Overall, TFB's best use case thus far is perhaps with KRLS, on high-dimensional data with a good number of observations, where the researcher suspects the true functional form is reasonably smooth. Fortunately, lots of further work is currently planned:

1. Expansion of the `tfb` package for use with many more regression methods (notably least absolute shrinkage and selection operator, LASSO), including the ability for the user to perform custom regressions without needing to specify the intended fit.

2. Implementation of arbitrary k-fold cross-fitting for the `tfb` package.

3. Generalization of TFB to nonparametric models, notably random forest, which may then be implemented into the package.

4. Documentation and publishing of the package for accessibility.

The following further work is also being considered, though it is not currently planned:

1. Finding a tighter upper bound on the worst case imbalance.

2. Programming TFB in other programming languages such as Python and C++, and potentially doing so without the usage of MOSEK, which limits the code's accessibility.

3. Generalization of TFB to continuous treatments.

4. The asymptotics of specification, namely to what extent the initial regression in TFB can be misspecified before the unbiasedness or efficiency of its estimator collapses.

5. The asymptotic normality of $\hat{\tau}_{\text{wdim}}$, specifically as relates to cross-fitting: determining the asymptotic correlation of estimates on different k-fold sample splits and their asymptotic variance if they are not asymptotically uncorrelated.

Appendix A: Rewriting the TFB Optimization Problem for the ATE for Use in RMosek

This minimization problem is theoretically good to go, but its objective function is rather complicated. However, its objective function happens to be convex. So we'll use RMosek to code it in R, a software specialized in convex optimization (MOSEK ApS, 2019). In order to do this, we need to represent the objective function linearly, using methodology as described in Koenker & Mizera (2014). We can do this by constructing an equivalent optimization problem with a linear function, but with many more parameters and domain restraints. While the linear and cone constraints set on the new parameters will ensure that the convex optimization gives equivalent results for the weights, we are not interested in their actual values and use them only to simplify the expression.

For the first substitution, we create linear constraint matrices

$$\begin{aligned} V_{\text{raw},d} &= \text{imbal}_d^{\text{ATE}}(w, g_d(X), D) \\ V_{\text{tf},d} &= \text{imbal}_d^{\text{ATE}}(w, g_d(X), D)\hat{V}_{\beta_d}^{\frac{1}{2}} \end{aligned}$$

These respectively refer to the raw imbalance of $g_d(X)$, and the imbalance of $g_d(X)$ transformed by $\hat{V}_{\beta_d}^{\frac{1}{2}}$. Substituting in, the equivalent optimization is:

$$\underset{V_{\text{tf},d}}{\overset{w,V_{\text{raw},d},}{\text{argmin}}} \left[\left(\sqrt{Q_q(\chi^2_{p_1})} \cdot \|V_{\text{tf},1}\|_2 + |V_{\text{raw},1}\hat{\beta}_1| + \sqrt{Q_q(\chi^2_{p_0})} \cdot \|V_{\text{tf},0}\|_2 + |V_{\text{raw},0}\hat{\beta}_0|\right)^2 \right.$$
$$\left. + \frac{\hat{\sigma}_0^2}{n_0^2} \sum_{i:\, D_i=0} w_i^2 + \frac{\hat{\sigma}_1^2}{n_1^2} \sum_{i:\, D_i=1} w_i^2 \right]$$

For the second substitution, we create linear terms

$$V_{\text{raw},d}\hat{\beta}_d = t^{(1)}_{\text{mag},d} - t^{(2)}_{\text{mag},d}$$

Each $t^{(i)}_{\text{mag},d}$ is nonnegative. Consider the following two cases:

$$\begin{aligned} & t^{(1)}_{\text{mag},d} := \left|V_{\text{raw},d}\hat{\beta}_d\right| && \text{case 1} \\ \implies & t^{(2)}_{\text{mag},d} = 0 \end{aligned}$$

$$\begin{aligned} & t^{(2)}_{\text{mag},d} := \left|V_{\text{raw},d}\hat{\beta}_d\right| && \text{case 2} \\ \implies & t^{(1)}_{\text{mag},d} = 0 \end{aligned}$$

Then at least one of these cases must be true for any real $V_{\text{raw},d}\hat{\beta}_d$. Regardless of the case(s) we necessarily use, it follows that

$$\min_{\substack{w, V_{\text{raw},d}, \\ V_{\text{tf},d}, t_{\text{mag},d}}} \left(t^{(1)}_{\text{mag},d} + t^{(2)}_{\text{mag},d}\right) = \min_{\substack{w, V_{\text{raw},d}, \\ V_{\text{tf},d}}} \left|V_{\text{raw},d}\hat{\beta}_d\right|$$

irrespective of the values of w, $V_{\text{raw},d}$, and $V_{\text{tf},d}$. Thus we can rewrite the optimization equivalently as

$$\underset{\substack{w, V_{\text{raw},d}, \\ V_{\text{tf},d}, t_{\text{mag},d}}}{\text{argmin}} \left[\left(\sqrt{Q_q(\chi^2_{p_1})} \cdot \|V_{\text{tf},1}\|_2 + (t^{(1)}_{\text{mag},1} + t^{(2)}_{\text{mag},1}) + \sqrt{Q_q(\chi^2_{p_0})} \cdot \|V_{\text{tf},0}\|_2 + (t^{(1)}_{\text{mag},0} + t^{(2)}_{\text{mag},0}) \right)^2 + \frac{\hat{\sigma}^2_0}{n^2_0} \sum_{i:\, D_i=0} w_i^2 + \frac{\hat{\sigma}^2_1}{n^2_1} \sum_{i:\, D_i=1} w_i^2 \right]$$

Next, we introduce quadratic cone constraints

$$\|V_{\text{tf},d}\| \leq t_{\text{quad},d}$$

where $t_{\text{quad},d} \geq 0$. The new terms minimize to the transformed imbalances we already have, so substituting in gives an equivalent optimization problem.

$$\underset{\substack{w_d, V_{\text{raw},d}, V_{\text{tf},d}, \\ t_{\text{mag},d}, t_{\text{quad},d}}}{\text{argmin}} \left[\left(\sqrt{Q_q(\chi^2_{p_1})} \cdot t_{\text{quad},1} + (t^{(1)}_{\text{mag},1} + t^{(2)}_{\text{mag},1}) + \sqrt{Q_q(\chi^2_{p_0})} \cdot t_{\text{quad},0} + (t^{(1)}_{\text{mag},0} + t^{(2)}_{\text{mag},0}) \right)^2 + \frac{\hat{\sigma}^2_0}{n^2_0} \sum_{i:\, D_i=0} w_i^2 + \frac{\hat{\sigma}^2_1}{n^2_1} \sum_{i:\, D_i=1} w_i^2 \right]$$

At this point, our large squared term can be simplified with the new linear constraint

$$\begin{aligned} t_{\text{bias}} = & \sqrt{Q_q(\chi^2_{p_1})} \cdot t_{\text{quad},1} + (t^{(1)}_{\text{mag},1} + t^{(2)}_{\text{mag},1}) \\ & + \sqrt{Q_q(\chi^2_{p_0})} \cdot t_{\text{quad},0} + (t^{(1)}_{\text{mag},0} + t^{(2)}_{\text{mag},0}) \end{aligned}$$

giving

$$\underset{t_{\text{mag},d}, t_{\text{quad},d}, t_{\text{bias}}}{\overset{w, V_{\text{raw},d}, V_{\text{tf},d},}{\text{argmin}}} \left[(t_{\text{bias}})^2 + \frac{\hat{\sigma}_0^2}{n_0^2} \sum_{i:\, D_i = 0} w_i^2 + \frac{\hat{\sigma}_1^2}{n_1^2} \sum_{i:\, D_i = 1} w_i^2 \right]$$

Next, we introduce the rotated quadratic cone constraints

$$\begin{aligned} t^2_{\text{bias}} &\leq 2u^{(1)}_a u^{(2)}_a \\ \sum_{i:\, D_i = d} w_i^2 &\leq 2u^{(1)}_{b,d} u^{(2)}_{b,d} \end{aligned}$$

Similar to the previous cone constraints, creating new terms that minimize to the problem we already have will not affect our optimization.

$$\underset{t_{\text{quad},d}, t_{\text{bias}}, u_a, u_{b,d}}{\overset{w, V_{\text{raw},d}, V_{\text{tf},d}, t_{\text{mag},d},}{\text{argmin}}} \left[\left(2u^{(1)}_a u^{(2)}_a\right) + \frac{\hat{\sigma}_0^2}{n_0^2} \left(2u^{(1)}_{b,0} u^{(2)}_{b,0}\right) + \frac{\hat{\sigma}_1^2}{n_1^2} \left(2u^{(1)}_{b,1} u^{(2)}_{b,1}\right) \right]$$

We can further create linear constraints

$$\begin{aligned} u^{(2)}_a &= \frac{1}{2} \\ u^{(2)}_{b,d} &= \frac{1}{2} \end{aligned}$$

This removes the factor of two currently present, and means our final problem will not have a squared term. The final optimization problem can then be stated as

$$\underset{t_{\text{quad},d}, t_{\text{bias}}, u_a, u_{b,d}}{\overset{w, V_{\text{raw},d}, V_{\text{tf},d}, t_{\text{mag},d},}{\text{argmin}}} \left[u^{(1)}_a + \frac{\hat{\sigma}_0^2}{n_0^2} u^{(1)}_{b,0} + \frac{\hat{\sigma}_1^2}{n_1^2} u^{(1)}_{b,1} \right]$$

with the list of restrictions:

Table A.1

bounds	dimensions	linear constraints	cone constraints
$0 \le w_0^{(i)} \le n_0$	$w_0 \in \mathbb{R}^{n_0}$	$\sum_{i:\ D_i=0} w_i = n_0$	$\lVert V_{\text{tf},0} \rVert \le t_{\text{quad},0}$
$0 \le w_1^{(i)} \le n_1$	$w_1 \in \mathbb{R}^{n_1}$	$\sum_{i:\ D_i=1} w_i = n_1$	$\lVert V_{\text{tf},1} \rVert \le t_{\text{quad},1}$
$-\infty \le V_{\text{raw},0} \le \infty$	$V_{\text{raw},0} \in \mathbb{R}^p$	$V_{\text{raw},0} = \text{imbal}_0^{\text{ATE}}(w, g_0(X), D)$	$t_{\text{bias}}^2 \le u_a^{(1)}$
$-\infty \le V_{\text{raw},1} \le \infty$	$V_{\text{raw},1} \in \mathbb{R}^p$	$V_{\text{raw},1} = \text{imbal}_1^{\text{ATE}}(w, g_1(X), D)$	$\sum_{i:\ D_i=0} w_0^{(i)} \le u_{b,0}^{(1)}$
$-\infty \le V_{\text{tf},0} \le \infty$	$V_{\text{tf},0} \in \mathbb{R}^p$	$V_{\text{tf},0} = \text{imbal}_0^{\text{ATE}}(w, g_0(X), D)\hat{V}_{\beta_0}^{\frac{1}{2}}$	$\sum_{i:\ D_i=1} w_1^{(i)} \le u_{b,1}^{(1)}$
$-\infty \le V_{\text{tf},1} \le \infty$	$V_{\text{tf},1} \in \mathbb{R}^p$	$V_{\text{tf},1} = \text{imbal}_1^{\text{ATE}}(w, g_1(X), D)\hat{V}_{\beta_1}^{\frac{1}{2}}$	
$0 \le t_{\text{mag},0} \le \infty$	$t_{\text{mag},0} \in \mathbb{R}^2$	$V_{\text{raw},0}^T \hat{\beta}_0 = t_{\text{mag},0}^{(1)} - t_{\text{mag},0}^{(2)}$	
$0 \le t_{\text{mag},1} \le \infty$	$t_{\text{mag},1} \in \mathbb{R}^2$	$V_{\text{raw},1}^T \hat{\beta}_1 = t_{\text{mag},1}^{(1)} - t_{\text{mag},1}^{(2)}$	
$0 \le t_{\text{quad},0} \le \infty$	$t_{\text{quad},0} \in \mathbb{R}$	$t_{\text{bias}} = \sqrt{Q_q(\chi_{p_1}^2)} \cdot t_{\text{quad},1} + (t_{\text{mag},1}^{(1)} + t_{\text{mag},1}^{(2)})$	
		$+\sqrt{Q_q(\chi_{p_0}^2)} \cdot t_{\text{quad},0} + (t_{\text{mag},0}^{(1)} + t_{\text{mag},0}^{(2)})$	
$0 \le t_{\text{quad},1} \le \infty$	$t_{\text{quad},1} \in \mathbb{R}$	$u_a^{(2)} = \frac{1}{2}$	
$0 \le t_{\text{bias}} \le \infty$	$t_{\text{bias}} \in \mathbb{R}$	$u_{b,0}^{(2)} = \frac{1}{2}$	
$0 \le u_a \le \infty$	$u_a \in \mathbb{R}^2$	$u_{b,1}^{(2)} = \frac{1}{2}$	
$0 \le u_{b,0} \le \infty$	$u_{b,0} \in \mathbb{R}^2$		
$0 \le u_{b,1} \le \infty$	$u_{b,1} \in \mathbb{R}^2$		

Appendix B: Description and Demonstration of Constituent TFB Functions

Let's give an overview of what TFB does at a lower level. Using the same data as before, we'll work through to reproduce the result given by the wrapper function `tfb()`.

To begin, TFB splits the given data in two. This is done with `split_sample()`, which takes the following input:

> `X_d`, two matrices $g_d(X)$, given explicitly or defined by the initial regression. In our example, these are identical as OLS does not require transformation of the covariates.
>
> `d`, the treatment vector D.
>
> `y`, the observed outcome vector Y.

Note that in input and output names, red `d` subscripts have been used for brevity and actually represent sets of names–the `d` can be substituted with `c` to represent $d = 0$, and `t` to represent $d = 1$. This produces the following output.

> `X_sd`, four matrices $g_d(X)$, subset by the two sample splits.
>
> `d_s`, two treatment vectors, subset by the two sample splits.
>
> `y_s`, two observed outcome vectors, subset by the two sample splits.
>
> `split_s_indices` two vectors of indices, one for each split.

In input and output names, blue `s` subscripts have been used for brevity and represent sets of names–they can be replaced with `1` to represent split 1, and `2` to represent split 2. Next, TFB performs up to four initial regressions, depending on the estimand. These are given by `ols_initial()` and `krls_initial()` respectively. For input they take

`X_s`, two matrices, one for each sample split.

`y_s`, two observed outcome vectors, subset by the two sample splits.

`d_s`, two treatment vectors, subset by the two sample splits.

`d`, whether the two subset matrices are in $g_0(X)$ or $g_1(X)$.

`params`, additional parameters for the regression function.

The initial regression is performed twice for up to two different treatment group targets, producing the output

`beta_s`, two coefficient vectors $\hat{\beta}$, trained on the two sample splits.

`V_s`, two estimated covariance matrices $\hat{V}_\beta$, for the coefficient vectors trained on the two sample splits.

`e_s`, two residual vectors, one for each sample split.

`sqrtV_s`, the square roots $\hat{V}_\beta^{\frac{1}{2}}$ of the estimated covariance matrices given above.

`sigma2_s`, two estimates of the variance in the model, one for each sample split.

`yhat_s`, two predicted outcome vectors, one for each sample split.

`X_s`, for KRLS only, two subsets of the kernel/gram matrix $K(X)$, one for each sample split. Necessary to reparametrize the regression space.

And, as a further note for when the initial regression method is KRLS, TFB includes a function `gram_matrix()` to recreate $K(X)$. Doing this manually is necessary because the code for KRLS standardizes covariates in order to keep the math involved simpler. But since we are training KRLS on subsets of X, the standardization won't actually be consistent between regressions. It takes the input

`X`, a scaled covariate matrix.

`b`, a bandwidth parameter equal to the number of covariates, determining the spread of the gaussian kernel.

and produces the output

`M`, the kernel/gram matrix of `X`.

After performing its initial regressions, TFB solves its optimization problem using `Rmosek`. For our example, this will be the linear objective function under the associated restrictions detailed in the derivation. For the ATT and ATC, this solution is found using `tfb_optimization_att()`, as the problems are equivalent and just performed on different treatment groups. This function will be run twice, once for each split, and it takes as input

`X`, a covariate matrix X.

`d`, a treatment vector D.

`beta`, a coefficient vector $\hat{\beta}$.

`sqrtV`, the square root of the estimated covariance matrix $\hat{V}_{\beta}^{\frac{1}{2}}$ of the coefficient vector given above.

`sigma2`, an estimate of the variance in the model.

`q`, the probability threshold q.

`rtol`, the `Rmosek` tolerance for optimization.

`verb`, the verbosity of the `Rmosek` output.

If the estimand is the ATE, the function for this will instead be `tfb_optimization_ate()` and some of the inputs will double in order to weight multiple treatment groups, becoming `X_d`, `beta_d`, `sqrtV_d`, and `sigma2_d` respectively. These functions then produce the output

`w`, a vector of weights w.

`status`, whether the optimization succeeded.

For the ATT and ATC, TFB can then use the weights to generate summary statistics through `tfb_summary_att()`. As input, this function takes

`y`, the observed response vector y.

`y_s`, two observed response vectors y, subset by each split.

`yhat_s`, two predicted response vectors $\hat{y}$, subset by each split.

`d`, the treatment vector d.

`d_s`, two treatment vectors d, subset by each split.

`w_s`, two vectors of weights w, one for each split.

`estimand`, the target for estimation.

`q`, the probability threshold q.

If the estimand is the ATE, the function is instead `tfb_summary_ate()` and it doubles the predicted outcome inputs to `yhat_sd`, but also no longer needs `estimand` as the argument is unambiguous. As output, these functions produce

`wdim_s`, two estimates $\hat{\tau}_{\text{wdim}}$, one for each sample split.

`wdim`, the mean of the previous two estimates.

`e_sd`, up to four residual vectors, one in each treatment group and split.

`v_hat_s`, two estimates of the variance of `wdim_s` respectively.

`v_hat`, the estimate of the variance of `wdim`.

`ci_lower`, the lower bound of a $q \cdot 100$ percent confidence interval.

`ci_upper`, the upper bound of a $q \cdot 100$ percent confidence interval.

`dim_s`, two unweighted $\hat{\tau}_{\text{dim}}$, one for each sample split.

`dim`, the unweighted $\hat{\tau}_{\text{dim}}$ in the entire dataset.

Finally, everything is compiled into the output of `tfb()` itself. If a user of the package wanted to make more extensive manual adjustments to the TFB method, they could use these helper functions in order to get the desired result. We'll replicate the first few weights in our dataset and the final estimate to demonstrate their usage.

```r
set.seed(121)
# sample splitting
out <- tfb:::split_sample(X, X, d, y)

names <- c("n","X_1c","X_2c","X_1t","X_2t","d_1","d_2",
            "y_1","y_2","split_1_indices","split_2_indices")

for (i in 1:length(names)) {
  assign(names[i], out[[i]])
}
```

```r
# initial regressions
out <- tfb:::ols_initial(X_1c, X_2c, y_1, y_2, d_1, d_2)

names <- c("beta_1c","beta_2c","V_1c","V_2c","e_1c",
             "e_2c","sqrtV_1c","sqrtV_2c","sigma2_1c",
             "sigma2_2c","yhat_1c","yhat_2c")

for (i in 1:length(names)) {
  assign(names[i], out[[i]])
}

out <- tfb:::ols_initial(X_1t, X_2t, y_1, y_2, d_1, d_2, d = 1)

names <- c("beta_1t","beta_2t","V_1t","V_2t","e_1t",
             "e_2t","sqrtV_1t","sqrtV_2t","sigma2_1t",
             "sigma2_2t","yhat_1t","yhat_2t")

for (i in 1:length(names)) {
  assign(names[i], out[[i]])
}
```

```r
# finding weights
# the beta hats and their associated variables have to be from the
# opposite sample for the estimate to be asymptotically normal
w_1 <- tfb:::tfb_optimization_ate(
    X_c = X_1c,
    X_t = X_1t,
    d = d_1,
    beta_c = beta_2c,
    beta_t = beta_2t,
    sqrtV_c = sqrtV_2c,
    sqrtV_t = sqrtV_2t,
    sigma2_c = sigma2_2c,
    sigma2_t = sigma2_2t,
    q = 0.95,
    rtol = 1e-16,
    verb = 0
)$w

w_2 <- tfb:::tfb_optimization_ate(
    X_c = X_2c,
    X_t = X_2t,
    d = d_2,
    beta_c = beta_1c,
    beta_t = beta_1t,
    sqrtV_c = sqrtV_1c,
    sqrtV_t = sqrtV_1t,
    sigma2_c = sigma2_1c,
    sigma2_t = sigma2_1t,
    q = 0.95,
    rtol = 1e-16,
    verb = 0
)$w
```

```
# summary statistics
wdim <- tfb:::tfb_summary_ate(y, y_1, y_2, yhat_1c, yhat_2c,
                              yhat_1t, yhat_2t, d, d_1, d_2,
                              w_1, w_2, 0.95)[[3]]
```

And after running all our helper functions manually, we see that we get the same expected results, including the same weights and estimate as when we used the wrapper function.

```
# making vector of weights
weight <- rep(0,200)
weight[split_1_indices] <- w_1
weight[split_2_indices] <- w_2

# taking the top 6 for comparison
data.frame(i = seq(1,200), weight = weight) %>%
  head()
```

Table B.1

i	weight
1	0.8884137
2	0.7455230
3	1.0820791
4	0.9360968
5	0.6921248
6	0.4604304

```r
tibble(wdim = wdim)
```

Table B.2

wdim
-0.4006913

Appendix C: Code

```r
# code chunk 3.1.1
# loads required packages

library(tfb)
library(tidyverse)
library(kbal)
library(KRLS)
```

```r
# code chunk 3.1.2
# dgp 1

makedf <- function(n = 200) {
  x1 <- rnorm(n)            # E 0, V 1
  x2 <- rnorm(n, sd = 0.25) # E 0, V 0.0625
  p <- exp(0.7*x1+x2)/(1+exp(0.7*x1+x2))
  d <- runif(n) < p
  e0 <- rnorm(n, 0, 2)
  e1 <- rnorm(n, 0, 1.5)
  y0 <- x1 + 2 * x2 + e0
  y1 <- x1 + 2 * x2 + 3 * x1 + x2 + e1
  y <- y0 * (1 - d) + y1 * d

  return(cbind(x1,x2,d,y,y0,y1))
}
```

```r
# code chunk 3.1.3
# some shorthand functions
```

```
satx <- function(d,y0,y1,estimand) {
  if (estimand == "atc") {
    d <- 1-d
  }
  if (estimand == "ate") {
    target <- d
  } else {
    target <- 1
  }
  return(mean(y1[d == target]) - mean(y0[d == target]))
}

wdim <- function(w,d,y,estimand) {
  if (estimand == "atc") {
    sign <- -1
  } else {
    sign <- 1
  }
  if (estimand == "ate") {
    w_ <- w
  } else {
    w_ <- 1
  }
  return(sign*(mean((w_ * y)[d == 1]) - mean((w * y)[d == 0])))
}

tfbwdim <- function(X,d,y,fit,estimand) {
  out <- tfb(X,d,y,fit,estimand)
  return(out$estimate$est)
}

kbalwdim <- function(X,d,y,estimand) {
  w <- rep(1,length(d))
  if (estimand == "atc") {
    d <- 1 - d
  }
  if (estimand != "ate") {
    out <- kbal(
      allx = X,
      treatment = d,
      meanfirst = T,
      printprogress = F
    )
    w <- out$w
  } else {
    out <- kbal(
      allx = X,
      sampled = d,
      sampledinpop = T,
      meanfirst = T,
      printprogress = F
    )
    w1 <- out$w
    out <- kbal(
      allx = X,
      sampled = 1-d,
      sampledinpop = T,
      meanfirst = T,
      printprogress = F
    )
    w0 <- out$w
    w[d == 0] <- w0[d == 0]
    w[d == 1] <- w1[d == 1]
  }
  estimate <- wdim(w,d,y,estimand)
  return(estimate)
}

estimate <- function(df) {
  estimates <- c()
  for (i in 1:6) {
    assign(
      c("w","X","d","y","y0","y1")[i],
      list(1,df[,1:2],df[,3],df[,4],df[,5],df[,6])[[i]]
    )
  }

  for (estimator in c(
    "satx_._att","satx_._atc","satx_._ate","wdim_._ate",
    "tfbwdim_lm_att","tfbwdim_lm_atc","tfbwdim_lm_ate",
    "tfbwdim_krls_att","tfbwdim_krls_atc","tfbwdim_krls_ate",
    "kbalwdim_._att","kbalwdim_._atc","kbalwdim_._ate"
  )) {

    for (i in 1:3) {
      assign(
        c("function_name","fit","estimand")[i],
        strsplit(estimator, "_")[[1]][i]
      )
    }

    args <- list(
      satx = list(d,y0,y1,estimand),
      wdim = list(w,d,y,estimand),
      tfbwdim = list(X,d,y,fit,estimand),
      kbalwdim = list(X,d,y,estimand)
    )

    function_args <- args[[function_name]]
    estimates <- c(
      estimates,do.call(function_name, function_args)
    )
  }
  return(matrix(estimates, nrow = 1))
}
```

```
# code chunk 3.1.4
# simulates dgp 1 and recovers estimates

n <- 200
iter <- 400
estimators <- matrix(nrow = 0, ncol = 13)

for (i in 1:iter) {
  set.seed(i)
  df <- makedf(n)
  sink("kbal_output.txt")
  estimators <- estimators %>%
    rbind(suppressWarnings(estimate(df)))
  sink()
  print(paste(
    "passed",as.character(i),"/",as.character(iter),
    "iterations of estimation for n =",
    as.character(n),sep=" "
  ))
}

closeAllConnections()

df <- data.frame(estimators)
colnames(df) <- c(
  "SATT","SATC","SATE","DIM",
  "TFB-OLS\nWDIM-ATT","TFB-OLS\nWDIM-ATC","TFB-OLS\nWDIM-ATE",
  "TFB-KRLS\nWDIM-ATT","TFB-KRLS\nWDIM-ATC","TFB-KRLS\nWDIM-ATE",
  "KBAL\nWDIM-ATT","KBAL\nWDIM-ATC","KBAL\nWDIM-ATE"
)

df %>%
  pivot_longer(cols = everything(),
               names_to = "stat", values_to = "val") %>%
  write.csv("lm_simulation.csv")
```

```r
# code chunk 3.1.5
# finding mean estimates

df <- read.csv("lm_simulation.csv") %>%
  mutate(att_error = 0, atc_error = 0, ate_error = 0)

for (estimator in unique(df$stat)) {
  assign(
    estimator,
    df %>%
      filter(stat == estimator) %>%
      pull(val) %>%
      mean()
  )
}

for (estimator in unique(df$stat)) {
  df[df$stat == estimator,"att_error"] <- abs(
    eval(as.name(estimator))-SATT
  )
  df[df$stat == estimator,"atc_error"] <- abs(
    eval(as.name(estimator))-SATC
  )
  df[df$stat == estimator,"ate_error"] <- abs(
    eval(as.name(estimator))-SATE
  )
}
```

```r
# code chunk 3.1.6
# making att boxplot

df %>%
  mutate(stat = fct_reorder(stat,att_error)) %>%
  filter(grepl("ATT",stat) | stat == "DIM") %>%
  ggplot(aes(x = stat, y = val, fill = stat)) +
  geom_boxplot() +
  geom_hline(yintercept = SATT) +
  scale_fill_manual(
    values = c("#648FFF","#785EF0","#DC267F","#FE6100","#FFB000")
  ) +
  labs(
    title = paste("Figure 3.1.1: Comparison of ATT Estimator",
                  "Performance on a Linear DGP",sep = " "),
    y = "Estimated Value",
    x = "Estimator"
  ) +
  theme(legend.position="none")
```

```
# code chunk 3.1.7
# making atc boxplot

df %>%
  mutate(stat = fct_reorder(stat,atc_error)) %>%
  filter(grepl("ATC",stat) | stat == "DIM") %>%
  ggplot(aes(x = stat, y = val, fill = stat)) +
  geom_boxplot() +
  geom_hline(yintercept = SATC) +
  scale_fill_manual(
    values = c("#648FFF","#785EF0","#DC267F","#FE6100","#FFB000")
  ) +
  labs(
    title = paste("Figure 3.1.2: Comparison of ATC Estimator",
                  "Performance on a Linear DGP",sep = " "),
    y = "Estimated Value",
    x = "Estimator"
  ) +
  theme(legend.position="none")
```

```
# code chunk 3.1.8
# making ate boxplot

df %>%
```

```r
  mutate(stat = fct_reorder(stat,ate_error)) %>%
  filter(grepl("ATE",stat) | stat == "DIM") %>%
  ggplot(aes(x = stat, y = val, fill = stat)) +
  geom_boxplot() +
  geom_hline(yintercept = SATE) +
  scale_fill_manual(
    values = c("#648FFF","#785EF0","#DC267F","#FE6100","#FFB000")
  ) +
  labs(
    title = paste("Figure 3.1.3: Comparison of ATE Estimator",
                  "Performance on a Linear DGP",sep = " "),
    y = "Estimated Value",
    x = "Estimator"
  ) +
  theme(legend.position="none")
```

```r
# code chunk 3.2.1
# dgp 2

makedf <- function(n) {
  x1 <- rchisq(n, df = 3) - 3 # E 0,   V 6
  x2 <- rbeta(n, 0.5, 0.5)    # E 0.5, V 0.125
  prob <- 1/(1+exp(x1*x2+0.5))
  d <- runif(n) < prob
  e0 <- rnorm(n, 0, 0.25)
  e1 <- rnorm(n, 0, 0.5)
  y0 <- (x1 - 3) * (x2 - 2) + e0
  y1 <- (x1 - 3) * (x1 + 3) * (x2 - 2) * (x2 + 3) + e1
  y <- y0 * (1 - d) + y1 * d

  return(cbind(x1,x2,d,y,y0,y1))
}
```

```r
# code chunk 3.2.2
# simulates dgp 2 and recovers estimates

```

```r
n <- 1000
iter <- 400
estimators <- matrix(nrow = 0, ncol = 13)

for (i in 1:iter) {
  set.seed(i)
  df <- makedf(n)
  sink("kbal_output.txt")
  estimators <- estimators %>%
    rbind(suppressWarnings(estimate(df)))
  sink()
  print(paste(
    "passed",as.character(i),"/",as.character(iter),
    "iterations of estimation for n =",
    as.character(n),sep=" "
  ))
}

closeAllConnections()

df <- data.frame(estimators)
colnames(df) <- c(
  "SATT","SATC","SATE","DIM",
  "TFB-OLS\nWDIM-ATT","TFB-OLS\nWDIM-ATC","TFB-OLS\nWDIM-ATE",
  "TFB-KRLS\nWDIM-ATT","TFB-KRLS\nWDIM-ATC","TFB-KRLS\nWDIM-ATE",
  "KBAL\nWDIM-ATT","KBAL\nWDIM-ATC","KBAL\nWDIM-ATE"
)

df %>%
  pivot_longer(cols = everything(),
                names_to = "stat", values_to = "val") %>%
  write.csv("krls_simulation.csv")
```

```r
# code chunk 3.2.3
# finding mean estimates
```

```
df <- read.csv("krls_simulation.csv") %>%
  mutate(att_error = 0, atc_error = 0, ate_error = 0)

for (estimator in unique(df$stat)) {
  assign(
    estimator,
    df %>%
      filter(stat == estimator) %>%
      pull(val) %>%
      mean()
  )
}

for (estimator in unique(df$stat)) {
  df[df$stat == estimator,"att_error"] <- abs(
    eval(as.name(estimator))-SATT
  )
  df[df$stat == estimator,"atc_error"] <- abs(
    eval(as.name(estimator))-SATC
  )
  df[df$stat == estimator,"ate_error"] <- abs(
    eval(as.name(estimator))-SATE
  )
}
```

```
# code chunk 3.2.4
# creating att boxplot

df %>%
  mutate(stat = fct_reorder(stat,att_error)) %>%
  filter(grepl("ATT",stat) | stat == "DIM") %>%
  ggplot(aes(x = stat, y = val, fill = stat)) +
  geom_boxplot() +
  geom_hline(yintercept = SATT) +
  scale_fill_manual(
    values = c("#648FFF","#FE6100","#DC267F","#785EF0","#FFB000")
  ) +
  labs(
    title = paste("Figure 3.2.1: Comparison of ATT Estimator",
                  "Performance on a Nonlinear DGP",sep = " "),
    y = "Estimated Value",
    x = "Estimator"
  ) +
  theme(legend.position="none")
```

```
# code chunk 3.2.5
# creating atc boxplot

df %>%
  mutate(stat = fct_reorder(stat,atc_error)) %>%
  filter(grepl("ATC",stat) | stat == "DIM") %>%
  ggplot(aes(x = stat, y = val, fill = stat)) +
  geom_boxplot() +
  geom_hline(yintercept = SATC) +
  scale_fill_manual(
    values = c("#648FFF","#DC267F","#785EF0","#FE6100","#FFB000")
  ) +
  labs(
    title = paste("Figure 3.2.2: Comparison of ATC Estimator",
                  "Performance on a Nonlinear DGP",sep = " "),
    y = "Estimated Value",
    x = "Estimator"
  ) +
  theme(legend.position="none")
```

```
# code chunk 3.2.6
# creating ate boxplot

df %>%
  mutate(stat = fct_reorder(stat,ate_error)) %>%
  filter(grepl("ATE",stat) | stat == "DIM") %>%
  ggplot(aes(x = stat, y = val, fill = stat)) +
  geom_boxplot() +
  geom_hline(yintercept = SATE) +
  scale_fill_manual(
    values = c("#648FFF","#DC267F","#785EF0","#FE6100","#FFB000")
  ) +
  labs(
    title = paste("Figure 3.2.3: Comparison of ATE Estimator",
                  "Performance on a Nonlinear DGP",sep = " "),
    y = "Estimated Value",
    x = "Estimator"
  ) +
  theme(legend.position="none")
```

```
# code chunk 3.2.7
# demonstrating the difficulty of DGP 2 for the ATC

set.seed(49)
df <- makedf(5000) %>%
  data.frame() %>%
  mutate(Group = if_else(d == 0,"Control","Treatment"))

df %>%
  ggplot(aes(x = x1, color = Group, fill = Group)) +
  geom_density(alpha = 0.1) +
  scale_color_manual(values = c("#648FFF","#DC267F")) +
  scale_fill_manual(values = c("#785EF0","#FE6100")) +
  labs(
    y = "Density",
    title = "Figure 3.2.4: Density of First Covariate
    in Control and Treatment Groups"
  )
```

```
# code chunk 3.2.8
# finding treatment group weights for TFB-OLS on the ATC

out <- tfb(as.matrix(df[,1:2]), df$d, df$y,
           fit = "lm", estimand = "atc")
w1 <- out$weights$weights_1
w2 <- out$weights$weights_2
w <- rep(1,5000)
w[out$indices$split_1][df$d[out$indices$split_1] == 1] <- w1
w[out$indices$split_2][df$d[out$indices$split_2] == 1] <- w2

df <- df %>%
  cbind(w)
```

```
# code chunk 3.2.9
# density plots after weighting for the ATC

df %>%
  ggplot(aes(x = w * x1, color = Group, fill = Group)) +
  geom_density(alpha = 0.1) +
  scale_color_manual(values = c("#648FFF","#DC267F")) +
  scale_fill_manual(values = c("#785EF0","#FE6100")) +
  labs(
    y = "Density",
    title = "Figure 3.2.5: Weighted Density of First Covariate
    in Control and Treatment Groups"
  )
```

```
# code chunk 3.2.10
# before and after weighting the treatment covariates for the ATC

df %>%
  filter(Group == "Treatment") %>%
  mutate(weighted = w * x1) %>%
  select(x1,weighted) %>%
  pivot_longer(cols = everything(), names_to = "stat", values_to = "val") %>%
  mutate(Weighting = if_else(stat == "x1","Before","After")) %>%
  mutate(Weighting = fct_relevel(Weighting,c("Before","After"))) %>%
  ggplot(aes(x = val, linetype = Weighting)) +
  geom_density(color = "#DC267F", fill = "#FE6100", alpha = 0.1) +
  labs(
    y = "Density",
    title = "Figure 3.2.6: Comparison of Weighted and Unweighted
    Distribution of First Covariate in Treatment Group"
  )
```

```
# code chunk 4.1.1
# required package installation

for (package in c("devtools","roxygen2","KRLS","SparseM")) {
  if (require(package) != T){
    install.packages(package)
    library(package)
  }
}

# Rmosek's installation is more complicated than the above packages.
# However, it is available for free with an educational license.
# See https://docs.mosek.com/latest/rmosek/install-interface.html
require(Rmosek)
```

```
# code chunk 4.1.2
# tfb installation

install("tfb")
library(tfb)
```

```
# code chunk 4.2.1
# observables of dgp 1

makedf <- function(n = 200) {
  x1 <- rnorm(n)              # E 0, V 1
  x2 <- rnorm(n, sd = 0.25) # E 0, V 0.0625
  p <- exp(0.7*x1+x2)/(1+exp(0.7*x1+x2))
  d <- runif(n) < p
  e0 <- rnorm(n, 0, 2)
  e1 <- rnorm(n, 0, 1.5)
  y0 <- x1 + 2 * x2 + e0
  y1 <- x1 + 2 * x2 + 3 * x1 + x2 + e1
  y <- y0 * (1 - d) + y1 * d

  return(cbind(x1,x2,d,y))
} # we can't observe y0 and y1 in practice
```

```
# code chunk 4.2.2
# generating a sample dataframe

set.seed(121)          # set a random seed so results are replicable
df <- makedf()         # using our dgp to create a matrix of data
df %>%                 # first 6 observations in our data
  data.frame() %>%
  head()
```

```
# code chunk 4.2.3
# running the tfb function on our data

X <- df[,1:2]
d <- df[,3]
y <- df[,4]

set.seed(121)
out <- tfb(
  X = X,             # the covariate matrix is given by
                     # the first two columns of our data
  d = d,             # the treatment vector is given by
                     # the third column of our data
  y = y,             # the observed response vector is given by
                     # the fourth column of our data
  fit = "lm",        # our specification of the initial regression
  estimand = "ate" # the estimand we want to estimate
)
```

```r
# code chunk 4.2.4
# showing the top of the dataframe with weights

# making vector of weights
weight <- rep(0,200)
weight[out$indices$split_1] <- out$weights$weights_1
weight[out$indices$split_2] <- out$weights$weights_2

# adding them to our data
cbind(weight,df) %>%
  data.frame() %>%
  head()
```

```r
# code chunk 4.2.5
# creating a table for estimator comparison

dim <- out$dim$dim
wdim <- out$estimate$est
tibble(
  stat = c("value","magnitude of bias"),
  dim = c(dim,abs(dim)),
  wdim = c(wdim,abs(wdim))
)
```

```r
# code chunk 4.2.6
# creating a table for the confidence interval

tibble(lower = out$ci$lower, upper = out$ci$upper)
```

```r
# code chunk 4.2.7
# displaying the p-value

tibble(p_val = out$p_val[[1]])
```

```
# code chunk 7.1.1
# splitting our data

set.seed(121)
# sample splitting
out <- tfb:::split_sample(X, X, d, y)

names <- c("n","X_1c","X_2c","X_1t","X_2t","d_1","d_2",
            "y_1","y_2","split_1_indices","split_2_indices")

for (i in 1:length(names)) {
  assign(names[i], out[[i]])
}
```

```
# code chunk 7.1.2
# regressing our data

# initial regressions
out <- tfb:::ols_initial(X_1c, X_2c, y_1, y_2, d_1, d_2)

names <- c("beta_1c","beta_2c","V_1c","V_2c","e_1c",
            "e_2c","sqrtV_1c","sqrtV_2c","sigma2_1c",
            "sigma2_2c","yhat_1c","yhat_2c")

for (i in 1:length(names)) {
  assign(names[i], out[[i]])
}

out <- tfb:::ols_initial(X_1t, X_2t, y_1, y_2, d_1, d_2, d = 1)

names <- c("beta_1t","beta_2t","V_1t","V_2t","e_1t",
            "e_2t","sqrtV_1t","sqrtV_2t","sigma2_1t",
            "sigma2_2t","yhat_1t","yhat_2t")

for (i in 1:length(names)) {
  assign(names[i], out[[i]])
}
```

```
# code chunk 7.1.3
# finding weights for our data

# finding weights
# the beta hats and their associated variables have to be from the
# opposite sample for the estimate to be asymptotically normal
w_1 <- tfb:::tfb_optimization_ate(
    X_c = X_1c,
    X_t = X_1t,
    d = d_1,
    beta_c = beta_2c,
    beta_t = beta_2t,
    sqrtV_c = sqrtV_2c,
    sqrtV_t = sqrtV_2t,
    sigma2_c = sigma2_2c,
    sigma2_t = sigma2_2t,
    q = 0.95,
    rtol = 1e-16,
    verb = 0
)$w

w_2 <- tfb:::tfb_optimization_ate(
    X_c = X_2c,
    X_t = X_2t,
    d = d_2,
    beta_c = beta_1c,
    beta_t = beta_1t,
    sqrtV_c = sqrtV_1c,
    sqrtV_t = sqrtV_1t,
    sigma2_c = sigma2_1c,
    sigma2_t = sigma2_1t,
    q = 0.95,
    rtol = 1e-16,
    verb = 0
```

```r
)$w
```

```r
# code chunk 7.1.4
# finding the estimate for our data

# summary statistics
wdim <- tfb:::tfb_summary_ate(y, y_1, y_2, yhat_1c, yhat_2c,
                              yhat_1t, yhat_2t, d, d_1, d_2,
                              w_1, w_2, 0.95)[[3]]
```

```r
# code chunk 7.1.5
# replicating the first few weights

# making vector of weights
weight <- rep(0,200)
weight[split_1_indices] <- w_1
weight[split_2_indices] <- w_2

# taking the top 6 for comparison
data.frame(i = seq(1,200), weight = weight) %>%
  head()
```

```r
# code chunk 7.1.6
# replicating our estimate

tibble(wdim = wdim)
```

References

Chernozhukov, V., Chetverikov, D., Demirer, M., Duflo, E., Hansen, C., Newey, W., & Robins, J. (2018). Double/debiased machine learning for treatment and structural parameters. *The Econometrics Journal*, *21*(1), C1–C68. http://doi.org/10.1111/ectj.12097

Grenander, U. (1954). On the estimation of regression coefficients in the case of an autocorrelated disturbance. *The Annals of Mathematical Statistics*, *25*(2), 252–272. Retrieved from `http://www.jstor.org/stable/2236729`

Hainmueller, J. (2012). Entropy balancing for causal effects: A multivariate reweighting method to produce balanced samples in observational studies. *Political Analysis*, *20*(1), 25–46. http://doi.org/10.1093/pan/mpr025

Hainmueller, J., & Hazlett, C. (2014). Kernel regularized least squares: Reducing misspecification bias with a flexible and interpretable machine learning approach. *Political Analysis*, *22*(2), 143–168. http://doi.org/10.1093/pan/mpt019

Hazlett. (2020). Kernel balancing: A flexible non-parametric weighting procedure for estimating causal effects. *Statistica Sinica*, *30*(3), 1155–1189. http://doi.org/10.5705/ss.202017.0555

Hoerl, A. E., & Kennard, R. W. (2000). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, *42*(1), 80–86. Retrieved from `http://www.jstor.org/stable/1271436`

Koenker, R., & Mizera, I. (2014). Convex optimization in r. *Journal of Statistical Software*, *60*(5), 1–23. http://doi.org/10.18637/jss.v060.i05

MOSEK ApS. (2019). *The r to MOSEK optimization interface.* Retrieved from `http://www.mosek.com/`

Rosenbaum, P. R., & Rubin, D. B. (1983). The central role of the propensity score in observational studies for causal effects. *Biometrika*, *70*(1), 41–55. http://doi.org/10.1093/biomet/70.1.41

Splawa-Neyman, J., Dabrowska, D. M., & Speed, T. P. (1990). On the application of probability theory to agricultural experiments. Essay on principles. Section

9. *Statistical Science*, *5*(4), 465–472. Retrieved from `http://www.jstor.org/stable/2245382`

Wainstein, L. (2022). Targeted function balancing. *arXiv Preprint arXiv:2203.12179*.

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., … Yutani, H. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, *4*(43), 1686. http://doi.org/10.21105/joss.01686

Wickham, H., Danenberg, P., Csárdi, G., & Eugster, M. (2023). *roxygen2: In-line documentation for r*. Retrieved from `https://roxygen2.r-lib.org/`