12-2021

MatchThem: Matching and Weighting after Multiple Imputation

Farhad Pishgar

Noah Greifer

Clémence Leyrat

Elizabeth Stuart

MatchThem:: Matching and Weighting after Multiple Imputation

by Farhad Pishgar, Noah Greifer, Clémence Leyrat and Elizabeth Stuart

Abstract Balancing the distributions of the confounders across the exposure levels in an observational study through matching or weighting is an accepted method to control for confounding due to these variables when estimating the association between an exposure and outcome and reducing the degree of dependence on certain modeling assumptions. Despite the increasing popularity in practice, these procedures cannot be immediately applied to datasets with missing values. Multiple imputation of the missing data is a popular approach to account for missing values while preserving the number of units in the dataset and accounting for the uncertainty in the missing values. However, to the best of our knowledge, there is no comprehensive matching and weighting software that can be easily implemented with multiply imputed datasets. In this paper, we review this problem and suggest a framework to map out the matching and weighting of multiply imputed datasets to 5 actions as well as the best practices to assess balance in these datasets after matching and weighting. We also illustrate these approaches using a companion package for R, **MatchThem**.

1. Introduction

Researchers often seek to estimate the effect of a treatment, exposure, or policy on an outcome but may be unable to randomly assign participants into the groups to be compared. The inability to randomize can lead to differences between the distributions of participant characteristics between exposure groups (known as *covariate imbalance*), which is the source of confounding bias in a naïve estimate of the exposure effect. When enough confounders—causes of exposure status and the outcome of interest—have been observed, one strategy for reducing this bias is to equate the confounder distributions across the exposure groups by matching or weighting units prior to estimating the exposure effect. Ideally, after matching or weighting, the exposure groups will be *balanced*, and a simple or covariate-adjusted estimate of the difference in average outcomes between the exposure groups will be unbiased for the true exposure effect (Stuart, 2010). Matching and weighting can also enhance the robustness to misspecification of any outcome models used to estimate the exposure effect (Ho et al., 2007).

Despite increasing popularity in practice, matching and weighting methods cannot be immediately applied to datasets with missing values. There are several solutions to address the problem of missing data in causal effect estimation, but a standard and relatively easy-to-use strategy is multiple imputation of the missing data, which preserves the number of units in the dataset while accounting for some of the uncertainty in the missing values (Cham and West, 2016). Multiple imputation involves filling in the missing data points using estimates of their values, repeating the process multiple times with randomness incorporated into each prediction to arrive at a set of multiple complete datasets containing the imputed values. The difficulty of analyzing multiply imputed data is that any analysis must be carried out within each imputed dataset, and the results pooled together using specific combining rules to arrive at a single set of estimates. Because matching and weighting are iterative, multi-step procedures, it is not straightforward to implement an analysis using these methods without extensive programming.

The **MatchThem** R package, which we introduce here, offers an analysis pipeline for estimating exposure effects using matching and weighting with multiply imputed data. The functions **MatchThem** offers blend seamlessly with functions used in other R packages for matching, weighting, and the generation and analysis of multiply imputed data. The aims of the present paper are to briefly review the issues around matching and weighting with multiply imputed data (section 2), to describe the structure and functionality of **MatchThem** (section 3), to describe the steps involved in implementing the best practices for these procedures (section 4), and to demonstrate the typical use of the **MatchThem** R package (section 5).

1.1. Notation

Let $i = 1, 2, 3, ..., n$ index the n units in a dataset, in which the causal effects of a binary exposure indicator (z) on an outcome (y) in the presence of a set of potential confounders ($X = \{x_1, x_2, x_3, ...\}$) are to be estimated (such that $z_i = 0$ indicates that unit i is assigned to the control group and $z_i = 1$ indicates that the unit i is assigned to the treated group) (Figure 1).

The typical context in which matching and weighting are used is one where data have been

collected from an observational study in which the exposure is not randomized, yielding systematic differences between exposed and unexposed units on a set of measured potential confounders (often referred to as *covariates*). The situation we consider here is one in which the values of some of the covariates or the outcome are missing for some units in the observed dataset (we do not consider situations in which the exposure status is missing as the methods described herein may not apply to such scenarios). In order to account for the missingness in the covariates and outcome, the missing values are multiply imputed, creating m complete datasets. Though we briefly explain the procedure of multiple imputation in section 2, here, we focus on the procedures following imputation; see (White et al., 2011) and (Azur et al., 2011) for accessible introductions to multiple imputation for medical researchers.

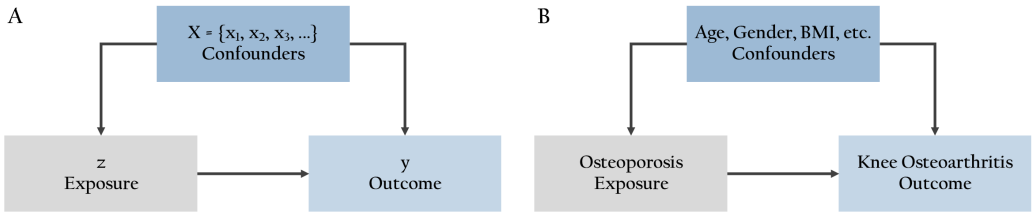

Figure 1: The Research Question. A. The notations used in this paper. B. The research question used as an example in this paper.

1.2. Software requirements

The **MatchThem** package works with the R statistical software and programming language and can be installed in R (with version $\geqslant$ 3.6.0) running on different platforms. **MatchThem** can be installed from the Comprehensive R Archive Network by executing the following commands in the R software console (the **MatchThem** package depends on the **MatchIt** (Ho et al., 2011) and **WeightIt** (Greifer, 2020a) packages; these lines will install those packages too):

```
install.packages("MatchThem")
library(MatchThem)
```

2. Matching, weighting, and missing data

2.1. Matching

Matching and weighting are methods to equate the distributions of the covariates between exposure groups (Stuart, 2010). Matching does so by duplicating, selecting, or dropping units from the dataset in such a way that the resulting exposure groups have similar covariate distributions. Typically, matching relies on a distance measure constructed from the covariates to pair similar units between exposure groups, which then form the resulting matched sample; a popular distance measure is the propensity score difference, the propensity score being the predicted probability of being in the exposed group given the covariates. Propensity scores can be used in a variety of matching algorithms (Ho et al., 2011; Williamson et al., 2012), though other distance measures can be used as well (as there have been some recent concerns about the use of propensity scores for matching (King and Nielsen, 2019)). Matching produces a set of matching weights (often 1 for those retained and 0 for those dropped) and matched pair membership, which can be incorporated into a regression of the outcome on the exposure to estimate the exposure effect in the matched sample. If the balance is achieved across the exposure groups in the matched sample, then bias in the exposure effect estimate will be reduced.

Matching is implemented in a number of R packages, but the **MatchIt** package provides access to a variety of matching methods for complete (i.e., non-missing) data. The **MatchIt** function `matchit()` performs the requested form of matching on the supplied dataset, returning an object from which matching weights and matched pair membership can be extracted for use in effect estimation. **MatchThem** interfaces with **MatchIt** to extend `matchit()` for use with multiply imputed data.

2.2. Weighting

Weighting is another way to achieve balance and reduce bias in the estimate of an exposure effect. Weights for each unit can be estimated so that the distributions of the covariates are the same across

the exposure groups in the weighted samples. The weights then function like survey weights and can be used in a weighted regression of the outcome on the exposure to estimate the exposure effect. A common way of estimating weights is to use a function of the estimated propensity score, a procedure known as inverse probability weighting (IPW), though there have been some developments that bypass estimating the propensity score to estimate the weights directly (Hainmueller, 2012; Zubizarreta, 2015).

The R package **WeightIt** implements a variety of weighting methods and functions similarly to **MatchIt**. The **WeightIt**'s function, `weightit()`, performs the requested form of weighting and returns an object containing the estimated weights. **MatchThem** interfaces with **WeightIt** to extend `weightit()` for use with multiply imputed data.

2.3. Assessing covariate balance

After matching or weighting, one must assess the degree to which the balancing method was successful at achieving covariate balance in the exposure groups. This involves using numerical and graphical criteria to compare the distributions of covariates across the groups (Austin, 2009). If the balance is not achieved, the matching or weighting specification should be changed, and the procedure performed again until a satisfactory balance is found. The **cobalt** package provides tools for assessing balance after matching and weighting and has tools for summarizing balance in multiply imputed data. **MatchThem** interfaces with **cobalt** to facilitate balance checking as part of a complete analysis pipeline. We refer readers to the **cobalt** documentation for further explanation of **cobalt**'s capabilities in order to maintain focus on the structure and functionality of **MatchThem**, but we will include the use of **cobalt** in the demonstration of the analysis pipeline in section 5.

2.4. Missing data

A major obstacle for most matching and weighting procedures is that they cannot be performed in a straightforward way for units with missing values in the covariates because these procedures either search control and treat groups for units with similar covariate values or rely on the predictions from a model with the covariates as the predictors (i.e., the propensity scores), which cannot be computed in the presence of missing data.

Complete-case analysis, i.e., excluding units with missing values in the potential confounders or outcome, is a simple and naïve approach for handling missing data. However, the complete-case analysis may not be a valid option in all instances; the assumption of missingness completely-at-random, described below, is required to justify complete-case analysis and is often violated in observational data. In addition, it is possible that dropping units with any missing values may yield a dataset with few remaining units (Pigott, 2001). Another approach is to replace the missing values with an arbitrary constant and include indicators for missingness as additional covariates in X, though this can also allow bias to remain (Knol et al., 2010). The preferred method to address the problem of missing data that preserves the number of units in the dataset and often yields unbiased effect estimates is to impute the missing values using multiple imputation (Leyrat et al., 2019).

Multiple imputation refers to the procedure of substituting the missing values with a set of plausible values that reflect the uncertainty in predicting the true unobserved values, which results in m imputed (filled-in) datasets (Sterne et al., 2009). Multiple imputation is justified when the mechanism behind the missingness is ignorable, i.e., given the observed data, units with missing data represent a random subset of the dataset ('missing-completely-at-random' in Rubin's language (Rubin, 1987)) or when the probability that a value is missing relies on values of other observed variables, but not on the missing value itself or unobserved factors ('missing-at-random' in Rubin's language (Rubin, 1987)).

Several multiple imputation methods are described in the literature, and multiple statistical packages can be used to generate multiply imputed data. The general framework of these methods is the same: impute the missing values to produce m datasets, analyze the imputed datasets separately, and pool the results obtained in each imputed dataset using standard combining rules to arrive at a single estimate for the sample (Sterne et al., 2009; Rubin, 1987). A popular method of multiple imputation is multiple imputation with chained equations (MICE), which involves iteratively fitting models to predict the missing values and is implemented in the **mice** R package. The **mice** package contains the functions `mice()` to impute the missing values, `with()` to run a supplied analysis model on each imputed dataset, and `pool()` to pool the results of the models to arrive at a single set of coefficient estimates and standard errors, facilitating the creation and analysis of multiply imputed data in a single analysis pipeline requiring minimal programming. We refer the reader to the **mice** documentation for further details (van Buuren and Groothuis-Oudshoorn, 2011).

2.5. Matching and weighting multiply imputed datasets

Given the limitations of conducting a complete-case analysis, multiply imputing missing data before matching or weighting has become a popular alternative for use with missing data. There has been some research examining the performance and optimal use of matching and weighting with multiply imputed data, with a focus on the correct sequence of actions involved. There are two main approaches that have been identified:

1. The *within* approach: In this approach, matching or weighting is performed within each imputed dataset, using the observed and imputed covariate values, and the exposure effects estimated in each of the m matched or weighted datasets are pooled together (Leyrat et al., 2019).
2. The *across* approach: In this approach, propensity scores are averaged across the imputed datasets, and, using this averaged measure, matching or weighting is performed in the imputed datasets. Finally, the estimated exposure effects obtained from analyzing the matched or weighted datasets are pooled together (Mitra and Reiter, 2016).

The across approach has been demonstrated to have inferior statistical performance as compared to the within approach in many common scenarios (Leyrat et al., 2019; de Vries and Groenwold, 2016), though early research favored its use (Mitra and Reiter, 2016). In particular, the across approach seems most effective when the outcomes are not used to impute the missing covariate values (de Vries and Groenwold, 2016). Although some recommend avoiding the inclusion of the outcome variable during or prior to matching and weighting with propensity scores (Rubin, 2001), statistical evidence favors the use of the outcome variable in multiple imputation of covariates (Leyrat et al., 2019). In addition, the across approach is not compatible with matching and weighting methods that do not involve propensity scores, such as coarsened exact matching (de Vries and Groenwold, 2016), Mahalanobis distance matching, and entropy balancing (Hainmueller, 2012). Both approaches are implemented in **MatchThem** to facilitate comparison between them, though the within approach is the default in **MatchThem** functions and is the approach we recommend.

It should be noted that the across approach described by Mitra and Reiter (2016) differs slightly from that described here; in their procedure, the averaged propensity scores are used to estimate the causal effect in a single dataset consisting of just the observed exposure and outcome values, which are assumed to be non-missing. The procedure described here is in the spirit of the original method but allows for the presence of imputed outcomes and the use of imputed covariates in the effect estimation. When there is no missingness in the outcome and covariates are not used in the effect estimation, the two versions of this approach coincide.

3. Package contents and structure

MatchThem provides functions and `S3` classes to facilitate the use of matching and weighting with multiply imputed data and the estimation of exposure effects and their uncertainty (i.e., standard errors), which requires special care when done with matched or weighted multiply imputed data. In particular, **MatchThem** extends the functionality of **MatchIt** and **WeightIt** for matching and weighting to multiply imputed data and the functionality of **mice** for the analysis of multiply imputed data to matched and weighted data. **MatchThem** provides wrappers for functions in these packages to create a smooth workflow requiring minimal programming. Table 1 contains a summary of the functions and classes contained in **MatchThem**.

The **MatchThem** functions `matchthem()` and `weightthem()` are wrappers for `MatchIt::matchit()` and `WeightIt::weightit()` that apply them to each imputed dataset, supplied in the form of a `"mids"` object, the output of a call to `mice::mice()`, which contains the multiply imputed datasets (`matchthem()` and `weightthem()` are also compatible with `"amelia"` objects from the **Amelia** package, but they are first transformed into `"mids"` objects before matching or weighting is performed on them). `matchthem()` and `weightthem()` apply the corresponding functions to the imputed datasets using the requested approach, storing the outputs along with the original imputed data in a `"mimids"` or `"wimids"` object, which extend `mice`'s `"mids"` class to additionally contain the matching and weighting output. The `"mimids"` and `"wimids"` classes have a number of methods that extend `mice`'s functions for analyzing `"mids"` objects; in particular, **MatchThem** offers `complete()`, `with()`, and `pool()`, which function similarly to their equivalents in **mice**. **MatchThem** also contains methods for `cbind()`, `print()`, `summary()`, and `plot()` with `"mimids"` and `"wimids"` objects. We describe the syntax of these functions below.

Function	Input	Output	Extends	Description
`matchthem()`	`mids` object	`mimids` object	`MatchIt::matchit()`	Performs the requested form of matching on the imputed datasets.
`weightthem()`	`mids` object	`wimids` object	`WeightIt::weightit()`	Performs the requested form of weighting on the imputed datasets.
`complete()`	`mimids` or `wimids` object	`data.frame`[1]	`mice::complete()`	Extracts one or more imputed datasets from the supplied input along with the outputs of the matching or weighting procedure.
`with()`	`mimids` or `wimids` object	`mimira` object	`mice::with()`	Runs the supplied analysis model on each imputed dataset, incorporating the outputs of the matching or weighting procedure.
`pool()`	`mimira` object	`mimipo` object[2]	`mice::pool()`	Pools the coefficients and standard errors estimated across the imputed datasets to a single set of coefficient and standard error estimates.

Table 1: Primary Functions in **MatchThem**. 1. `complete()` can also produce outputs in other forms. 2. `mimipo` objects can be further analyzed by functions in **mice** as if they had come from `mice::pool()`.

3.1. matchthem()

The syntax for matchthem() is as follows:

```
matchthem(formula, datasets,
          approach = "within",
          method = "nearest",
          distance = "logit",
          distance.options = list(),
          discard = "none",
          reestimate = FALSE, ...)
```

The formula argument corresponds to the model formula, which relates the exposure (on the left-hand side) to the covariates. The datasets argument corresponds to the "mids" or "amelia" object containing the multiply imputed datasets. The approach argument, with options "within" and "across", corresponds to the approach used. The method argument corresponds to the method of matching used, which, as of now, can be one of the nearest neighbor matching ("nearest"), optimal full matching ("full"), propensity score subclassification ("subclass"), optimal pair matching ("optimal"), exact matching ("exact"), coarsened exact matching ("cem"), and genetic matching ("genetic"). The distance argument corresponds to the method used to define the distances between units; it can be "mahalanobis" for Mahalanobis distance matching or a method of estimating propensity scores (the default, "glm", estimates propensity scores using logistic regression). Note that only the methods that involve propensity scores are allowed with the across approach; as of now, these include "nearest", "full", "subclass", "optimal", and "genetic", and only when propensity scores are requested. The other arguments, including those supplied to ..., control other aspects of the matching procedure and are, along with formula, method, and distance, passed directly to matchit().

3.2. weightthem()

The syntax for weightthem() is as follows:

```
weightthem(formula, datasets,
           approach = "within",
           method = "ps", ...)
```

The formula, datasets, and approach arguments have the same meaning as those for matchthem(). The method argument controls the weighting method used, which, as of now, can be one of logistic regression propensity score weighting ("ps"), covariate balancing propensity score weighting ("cbps") and its nonparametric variety ("npcbps"), generalized boosted modeling propensity score weighting ("gbm"), entropy balancing ("ebal"), SuperLearner propensity score weighting ("super"), optimization-based weighting ("optweight"), energy balancing ("energy"), and Bayesian additive regression tree propensity score weighting ("bart"). Note that only methods that involve propensity scores can be used with the across approach; as of now, these include "ps", "cbps", "gbm", "super", and "bart". Arguments supplied to ... are passed to weightit() to control details of the weight estimation.

3.3. complete()

The syntax for complete.mimids() is as follows (the syntax for complete.wimids() is identical):

```
complete.mimids(data, action = 1,
                include = FALSE, mild = FALSE, all = TRUE, ...)

complete.wimids(data, action = 1,
                include = FALSE, mild = FALSE, all = TRUE, ...)
```

complete() extracts the multiply imputed and matched or weighted datasets from a "mimids" or "wimids" object, yielding a "data.frame" containing the imputed data and the outputs of the matching or weighting function. These additional outputs include the estimated weights (which are produced with both matching and weighting), matched pair membership, and estimated propensity scores (if used). This function extends complete.mids() from the **mice** package. The data argument corresponds to a "mimids" or "wimids" object, the output of a call to matchthem() or weightthem(). The action argument controls the format of the output; it can be supplied as a number to extract a single dataset corresponding to that imputation number or a string, such as "long", to produce an object (of one of several types) containing all of the imputed datasets; the mild argument provides

another way to control output. The `all` argument controls whether all units should be included in the output or just units with a weight greater than zero (i.e., units that have not been discarded or that were left unmatched). Though the datasets produced by `complete()` can be analyzed separately and the estimates manually combined using the appropriate combining rules, the `with()` and `pool()` methods facilitate proper analysis of the matched or weighted imputed data.

3.4. `with()`

The syntax for `with.mimids()` and `with.wimids()` are as follows:

```
with.mimids(data, expr, cluster, ...)

with.wimids(data, expr, ...)
```

The `with()` method for `"mimids"` and `"wimids"` objects extends the `with()` method for `"mids"` objects provided in **mice**. It works by extracting the imputed datasets one-by-one along with the matching or weighting outputs and applies the modeling function supplied to `expr` (e.g., a call to `glm()`, `survey::svyglm()`, or `survival::coxph()` with the outcome model formula included) to each of the datasets. No data argument needs to be supplied to the modeling function because the imputed datasets are automatically supplied by `with()`. `with()` automatically incorporates the estimated weights in the estimation function when possible. The output of a call to with is a `"mimira"` object, which contains the outputs of the models run on each imputed dataset and extends the corresponding `"mira"` class from **mice**.

In general, for generalized linear outcome models, the `svyglm()` function in the **survey** package should be used as it correctly accounts for the weights and produces approximately correct standard errors, whereas the standard errors resulting from a normal call to `glm()` will be inaccurate. `with()` automatically constructs and supplies the `svydesign` object containing the data and weights, so neither need to be supplied. When matching is used, and a modeling function from the **survey** package is supplied, information about pair membership is also supplied to appropriately account for the clustering in the standard error estimation as recommended by (Austin, 2011) (this functionality can be controlled using the cluster argument).

3.5. `pool()`

The syntax for `pool()` is as follows:

```
pool(object, dfcom = NULL)
```

`pool()` takes in a `"mimira"` object (supplied to the `object` argument) to pool the models and provide a single set of coefficient estimates and information required to compute their standard errors. The `dfcom` argument controls the degrees of freedom used for the tests of the coefficients and confidence intervals, which typically is close to the number of units in the original dataset. Because matching and weighting can yield datasets with different numbers of units remaining, the default is to use the smallest degrees of freedom from the supplied models if possible; otherwise, a large value is used to approximate a z-test. **MatchThem** re-exports `pool()` as a generic with methods for `"mira"` objects (the output of `mice::with.mids()`) and `"mimira"` objects as `mice::pool()` is not a generic. The output of `MatchThem::pool.mimira()` is a `"mimipo"` object, which can be used with the methods available in **mice** for `"mipo"` objects (e.g., `summary()` and `print()`).

3.6. Methods for `"mimids"` and `"wimids"` objects

MatchThem also contains methods for `cbind()`, `print()`, `summary()`, and `plot()` with `"mimids"` and `"wimids"` objects. `cbind()` allows one to add variables from an external dataset, not included in the original `"mids"` object, that one might wish to be involved in the effect estimation model, such as an outcome not involved in the imputation or a variable collected after the multiple imputation occurred. The `print()`, `summary()`, and `plot()` methods simply apply the corresponding function to the `matchit` or `weightit` objects contained within the `"mimids"` or `"wimids"` object. The **MatchIt** and **WeightIt** documentation details their functionality. An additional argument, `n`, determines to which imputed dataset the function should be applied (e.g., `print.mimids(x,n = 1)` prints the output of the call to `matchit()` used on the first imputed dataset).

4. Suggested workflow

The suggested workflow for pre-processing imputed datasets with matching or weighting and then analyzing them to estimate exposure effects using **MatchThem** is as follows (Figure 2):

1. **Imputing the Missing Data in the Dataset**: Data are imputed using functions in **mice** or **Amelia**. Data imputed using another package can be coerced to a `"mids"` object by the **mice** function `as.mids()` for use with **MatchThem** functions (van Buuren and Groothuis-Oudshoorn, 2011; Honaker et al., 2011).
2. **Matching or Weighting the Imputed Datasets**: Matching or weighting are performed using `matchthem()` or `weightthem()`, respectively, on the imputed datasets. The functions perform the matching or weighting within each imputed dataset using the specified approach.
3. **Assessing Balance on the Matched or Weighted Datasets**: Functions in the **cobalt** R package can be used to assess the balance to ensure that the resulting bias is small across imputed datasets. The `bal.tab()` and `love.plot()` functions in the **cobalt** package can be used directly on the output of `matchthem()` and `weightthem()` (Greifer, 2020b). If the balance is not achieved, step 2 should be repeated with different approaches or methods until it is.
4. **Analyzing the Matched or Weighted Datasets**: Using `with()` function from **MatchThem** package, causal effects and their standard errors are estimated in each of the matched or weighted imputed datasets. Robust standard errors should be used with weighting and most matching methods and are available through integration with the **survey** package (Lumley, 2004).
5. **Pooling the Causal Effect Estimates**: The `pool()` function from the package is used to pool the obtained causal effect estimates and standard errors from each dataset using Rubin's rules.

5. Example

In this section, we review the suggested workflow for matching and weighting multiply imputed datasets, using an example. The research question in this context is whether osteoporosis at baseline is associated with increased odds of developing knee osteoarthritis in the follow-up or not (Figure 1). We will use the `osteoarthritis` dataset (included in the **MatchThem** package):

```
library(MatchThem)
data('osteoarthritis')
```

The `osteoarthritis` dataset contains data on 7 characteristics (age: `AGE`, gender: `SEX`, body mass index: `BMI`, racial background: `RAC`, smoking status: `SMK`, osteoporosis at baseline: `OSP`, and knee osteoarthritis in the follow-up: `KOA`) of 2,585 individuals. The dataset contains missing data in `BMI`, `RAC`, `SMK`, and `KOA` variables. We assume the missing values are missing at random, justifying the use of multiple imputation.

```
summary(osteoarthritis)
```

5.1. Imputing the missing data in the dataset

We use the **mice** package to impute the missing data in the `osteoarthritis` dataset. See the **mice** package reference manual for more details about this step (van Buuren and Groothuis-Oudshoorn, 2011). We use 5 imputations for illustration, though more imputations are always better.

```
library(mice)
imputed.datasets <- mice(osteoarthritis, m = 5)
```

The code above produces the 5 imputed datasets and saves them in the `imputed.datasets` object (of class `"mids"`). This object will be supplied to **MatchThem** functions to perform matching and weighting in the imputed datasets.

5.2. Matching and weighting the imputed datasets

5.2.1. Matching the imputed datasets

In this example, we use `matchthem()` to match the multiply imputed datasets, `imputed.datasets`, using the `"within"` matching approach with nearest neighbor matching on the propensity score,

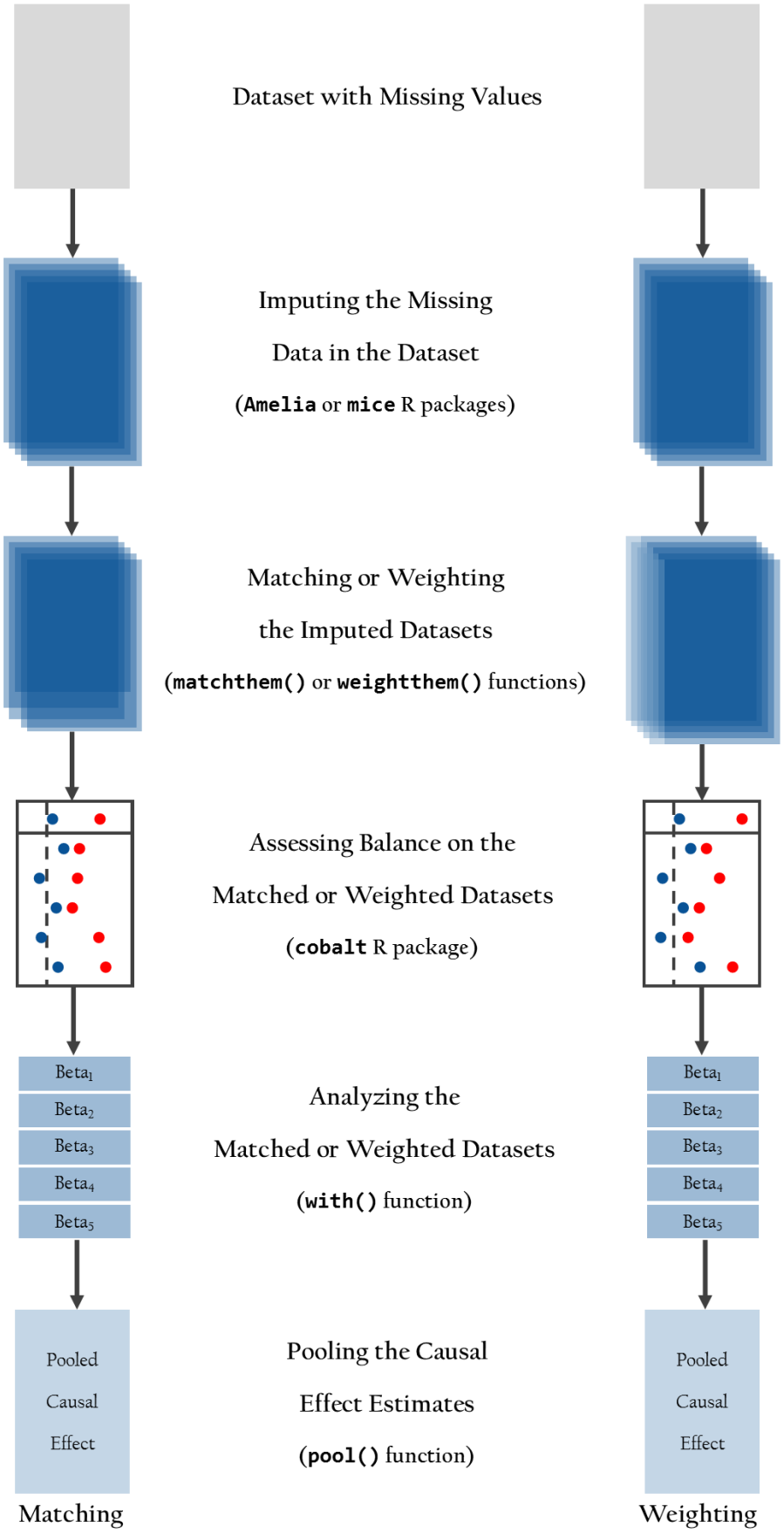

Figure 2: Suggested Workflow for Matching and Weighting Multiply Imputed Datasets

a caliper of 5% of the standard deviation of the propensity score, and 2:1 unexposed-to-exposed matching ratio for matching. The syntax is the same as it is for `MatchIt::matchit()`, except that the `imputed.datasets` is supplied to the `datasets` argument (whereas `matchit()` takes a `"data.frame"` for its `data` argument) and an argument to `approach` is supplied to select the approach to be used.

```
matched.datasets <- matchthem(OSP ~ AGE + SEX + BMI + RAC + SMK,
                              datasets = imputed.datasets,
                              approach = 'within',
                              method = 'nearest',
                              caliper = 0.05,
                              ratio = 2)

# Matching Observations  | dataset: #1 #2 #3 #4 #5
```

After the matching is performed in the 5 imputed datasets, the output will be saved in the `matched.datasets` object (of `"mimids"` class). The `"mimids"` object contains the original imputed data and the output of the calls to `matchit()` applied to each imputed dataset.

5.2.2. Weighting the imputed datasets

Here, we use `weightthem()` to weight the imputed datasets, `imputed.datasets`, using the `"across"` weighting approach with logistic regression propensity score weighting targeting the average treatment effect in the matched sample (ATM) estimand (which mimics the target population resulting from matching with a caliper (Li and Greene, 2013)). The syntax is the same as it is for `WeightIt::weightit()` except that the `imputed.datasets` is supplied to the `datasets` argument (whereas `weightit()` takes a `"data.frame"` for its `data` argument) and an argument to `approach` is supplied to select the approach to be used.

```
weighted.datasets <- weightthem(OSP ~ AGE + SEX + BMI + RAC + SMK,
                                datasets = imputed.datasets,
                                approach = 'across',
                                method = 'ps',
                                estimand = 'ATM')

# Estimating distances   | dataset: #1 #2 #3 #4 #5
# Estimating weights     | dataset: #1 #2 #3 #4 #5
```

The `weighted.datasets` object (of `"wimids"` class) contains the original imputed data and the output of the calls to `weightit()` applied to each imputed dataset.

5.3. Assessing balance on the matched and weighted datasets

Functions in the **cobalt** package are compatible with `"mimids"` and `"wimids"` objects, and the degree to which balance was achieved in the matched and weighted datasets of these objects can be assessed using the **cobalt** functions `bal.tab()`, `bal.plot()`, and `love.plot()`. Here, we illustrate the use of `bal.tab()` to compute absolute standardized mean differences (ASMDs) and Kolmogorov-Smirnov (KS) statistics for each covariate. The code below produces the largest ASMD and KS statistic after matching for each covariate across all the imputed datasets, indicating the worst balance across the datasets (Greifer, 2020b).

```
library(cobalt)
bal.tab(matched.datasets, stats = c('m', 'ks'),
        imp.fun = 'max')

# Balance summary across all imputations
#              Type Max.Diff.Adj Max.KS.Adj
# distance Distance       0.0107     0.0300
# AGE       Contin.       0.0296     0.0450
# SEX_2      Binary       0.0021     0.0021
# BMI       Contin.       0.0333     0.0557
# RAC_0      Binary       0.0021     0.0021
# RAC_1      Binary       0.0118     0.0118
# RAC_2      Binary       0.0139     0.0139
# RAC_3      Binary       0.0021     0.0021
```

```
# SMK        Binary       0.0128     0.0128

# Average sample sizes across imputations
#                      Control Treated
# All                  2106.     479.
# Matched (ESS)         738.17   467.2
# Matched (Unweighted)  810.2    467.2
# Unmatched            1295.8     11.8
```

This information shows that the covariates are well-balanced in the osteoporosis negative and positive groups after matching as the largest ASMD and KS statistics for all covariates across the imputed datasets are close to zero. The sample size information below indicates that some units were left unmatched and dropped from the sample. The displayed values are averaged across the imputed datasets.

We can produce the same balance table for the weighted datasets:

```
bal.tab(weighted.datasets, stats = c('m', 'ks'),
        imp.fun = 'max')

# Balance summary across all imputations
#                Type Max.Diff.Adj Max.KS.Adj
# prop.score Distance       0.0040     0.0433
# AGE         Contin.       0.0209     0.0376
# SEX_2        Binary       0.0002     0.0002
# BMI         Contin.       0.0070     0.0466
# RAC_0        Binary       0.0002     0.0002
# RAC_1        Binary       0.0019     0.0019
# RAC_2        Binary       0.0027     0.0027
# RAC_3        Binary       0.0006     0.0006
# SMK          Binary       0.0138     0.0138

# Average effective sample sizes across imputations
#            Control Treated
# Unadjusted 2106.     479.
# Adjusted    954.44   478.3
```

As with the matched datasets, the covariates are well balanced in the weighted datasets as demonstrated by the low values of the largest ASMD and KS statistics across the datasets. For more information on the available options for assessing balance with multiply imputed data, we refer the reader to the **cobalt** documentation (Greifer, 2020b).

5.4. Analyzing the matched and weighted datasets

The exposure effect within each imputed dataset can be estimated using `with()`, which applies the supplied outcome model to each of the matched or weighted datasets and stores the output in a `"mimira"` object. We illustrate the use of `with()` below to estimate the difference in the log odds of knee osteoarthritis between osteoporosis groups in the matched imputed datasets:

```
library(survey)
matched.models <- with(matched.datasets,
                       svyglm(KOA ~ OSP, family = quasibinomial()),
                       cluster = TRUE)
```

The models fit in each matched dataset are saved in the `matched.models` object (of `"mimira"` class). We can run the same code with the weighted imputed datasets:

```
weighted.models <- with(weighted.datasets,
                        svyglm(KOA ~ OSP, family = quasibinomial()))
```

Results are saved in the `weighted.models` object (of `"mimira"` class, note that in the calls to `svyglm()`, no `svydesign()` or `weights` arguments need to be specified as these are automatically supplied by `with()`)

5.5. Pooling the causal effect estimates

In order to arrive at a single set of coefficient and standard error estimates from the imputed datasets, we must pool the estimated models using `pool()`. We demonstrate this below on the `matched.models` object containing the models we fit above.

```
matched.results <- pool(matched.models)
```

The output of the `pool()` is saved in the `matched.results` object, which is of `"mimipo"` class. We can run `summary()` to arrive at a final set of estimates for the matched data:

```
summary(matched.results, conf.int = TRUE)
```

```
#          term   estimate  std.error  statistic       df  p.value    2.5 %    97.5 %
# 1 (Intercept) -0.2045680 0.08781317 -2.3295816 47.88960 0.024092 -0.38114 -0.027997
# 2        OSP1 -0.1255041 0.14138543 -0.8876733 53.09776 0.378720 -0.40908  0.158067
```

The displayed results show that there is not sufficient evidence of an association between osteoporosis and knee osteoarthritis development in the follow-up in this sample (beta = -0.13 [-0.41 – 0.16], odds ratio = 0.88 [0.66 – 1.17]).

We can run `pool()` and then `summary()` on the model fits in the weighted datasets to arrive at a similar table of results:

```
weighted.results <- pool(weighted.models)
summary(weighted.results, conf.int = TRUE)
```

```
#          term   estimate  std.error statistic      df    p.value    2.5 %    97.5 %
# 1 (Intercept) -0.1602278 0.06932782 -2.311162 225.930 0.02172525 -0.29684 -0.023616
# 2        OSP1 -0.1817342 0.13059573 -1.391579  63.768 0.16888557 -0.44265  0.079179
```

Again, there is no evidence for an association between osteoporosis and knee osteoarthritis development in this sample.

6. Summary

Matching and weighting are popular methods used to balance the distributions of potential confounders across the exposure levels in an observational study. However, these procedures cannot be immediately applied to datasets with missing values. Multiple imputation of the missing data can be an effective approach to account for missing values while preserving the number of units in the dataset and accounting for the uncertainty in the imputation of the missing values. In this paper, we described the functionality of **MatchThem**, which interfaces with **MatchIt**, **WeightIt**, and **mice** to provide a smooth, straightforward workflow for estimating exposure effects in multiply imputed data using matching or weighting. **MatchThem** contains functions and classes that encourage the use of best practices in analyzing matched or weighted multiply imputed data without requiring extensive programming by the user. Given the ubiquity of missing data in observational studies, we hope **MatchThem** will facilitate smart choices and reliable analyses by researchers attempting to estimate causal effects from observational data.

Bibliography

P. C. Austin. Some methods of propensity-score matching had superior performance to others: Results of an empirical investigation and monte carlo simulations. *Statistics in Medicine*, 51(1):171–184, 2009. URL https://doi.org/10.1002/sim.4200. [p294]

P. C. Austin. Comparing paired vs non-paired statistical methods of analyses when making inferences about absolute risk reductions in propensity-score matched samples. *Statistics in Medicine*, 30(11): 1292–1301, 2011. URL https://doi.org/10.1002/sim.4200. [p298]

M. J. Azur, E. A. Stuart, C. Frangakis, and P. J. Leaf. Multiple imputation by chained equations: What is it and how does it work? *International Journal of Methods in Psychiatric Research*, 20(1):40–49, 2011. URL https://doi.org/10.1002/mpr.329. [p293]

H. Cham and S. G. West. Propensity score analysis with missing data. *Psychological Methods*, 21(3): 427–445, 2016. URL https://doi.org/10.1037/met0000076. [p292]

B. P. de Vries and R. Groenwold. Comments on propensity score matching following multiple imputation. 25(6):3066—-3068, 2016. URL https://doi.org/10.1177/0962280216674296. [p295]

N. Greifer. *WeightIt: Weighting for Covariate Balance in Observational Studies*, 2020a. URL https://CRAN.R-project.org/package=WeightIt. R package version 0.9.0. [p293]

N. Greifer. *cobalt: Covariate Balance Tables and Plots*, 2020b. URL https://CRAN.R-project.org/package=cobalt. R package version 4.2.2. [p299, 301, 302]

J. Hainmueller. Entropy balancing for causal effects: A multivariate reweighting method to produce balanced samples in observational studies. *Political Analysis*, 20(1):25–46, 2012. URL https://doi.org/10.1093/pan/mpr025. [p294, 295]

D. E. Ho, K. Imai, G. King, and E. A. Stuart. Matching as nonparametric preprocessing for reducing model dependence in parametric causal inference. *Political Analysis*, 15(3):199–236, 2007. URL https://doi.org/10.1093/pan/mpl013. [p292]

D. E. Ho, K. Imai, G. King, and E. A. Stuart. MatchIt: Nonparametric preprocessing for parametric causal inference. *Journal of Statistical Software*, 42(8):1–28, 2011. URL https://doi.org/10.18637/jss.v042.i08. [p293]

J. Honaker, G. King, and M. Blackwell. Amelia II: A program for missing data. *Journal of Statistical Software*, 45(7):1–47, 2011. URL https://doi.org/10.18637/jss.v045.i07. [p299]

G. King and R. Nielsen. Why propensity scores should not be used for matching. *Political Analysis*, 27 (4):435–454, 2019. URL https://doi.org/10.1017/pan.2019.11. [p293]

M. J. Knol, K. J. M. Janssen, A. R. T. Donders, A. C. G. Egberts, E. R. Heerdink, D. E. Grobbee, K. G. M. Moons, and M. I. Geerlings. Unpredictable bias when using the missing indicator method or complete case analysis for missing confounder values: An empirical example. *Journal of Clinical Epidemiology*, 63(7):728–736, 2010. URL https://doi.org/10.1016/j.jclinepi.2009.08.028. [p294]

C. Leyrat, S. R. Seaman, I. R. White, I. Douglas, L. Smeeth, J. Kim, M. Resche-Rigon, J. R. Carpenter, and E. J. Williamson. Propensity score analysis with partially observed covariates: How should multiple imputation be used? *Statistical Methods in Medical Research*, 28(1):3–19, 2019. URL https://doi.org/10.1177/0962280217713032. [p294, 295]

L. Li and T. Greene. A weighting analogue to pair matching in propensity score analysis. *The International Journal of Biostatistics*, 9(2):215–234, 2013. URL https://doi.org/10.1515/ijb-2012-0030. [p301]

T. Lumley. Analysis of complex survey samples. *Journal of Statistical Software, Articles*, 9(8):1–19, 2004. URL https://doi.org/10.18637/jss.v009.i08. [p299]

R. Mitra and J. P. Reiter. A comparison of two methods of estimating propensity scores after multiple imputation. *Statistical Methods in Medical Research*, 25(1):188–204, 2016. URL https://doi.org/10.1177/0962280212445945. [p295]

T. D. Pigott. A review of methods for missing data. *Educational Research and Evaluation*, 7(4):353–383, 2001. URL https://doi.org/10.1076/edre.7.4.353.8937. [p294]

D. B. Rubin. *Multiple Imputation for Nonresponse in Surveys*, volume 81. John Wiley & Sons, 1987. [p294]

D. B. Rubin. Using propensity scores to help design observational studies: Application to the tobacco litigation. *Health Services and Outcomes Research Methodology*, 2(3-4):169–188, 2001. URL https://doi.org/10.1023/A:1020363010465. [p295]

J. A. Sterne, I. R. White, J. B. Carlin, M. Spratt, P. Royston, M. G. Kenward, A. M. Wood, and J. R. Carpenter. Multiple imputation for missing data in epidemiological and clinical research: Potential and pitfalls. *BMJ*, 338, 2009. URL https://doi.org/10.1136/bmj.b2393. [p294]

E. A. Stuart. Matching methods for causal inference: A review and a look forward. *Statistical Science*, 25(1):1–21, 2010. URL https://doi.org/10.1214/09-STS313. [p292, 293]

S. van Buuren and K. Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in R. *Journal of Statistical Software*, 45(3):1–67, 2011. URL https://doi.org/10.18637/jss.v045.i03. [p294, 299]

I. R. White, P. Royston, and A. M. Wood. Multiple imputation using chained equations: Issues and guidance for practice. *Statistics in Medicine*, 30(4):377–399, 2011. URL https://doi.org/10.1002/sim.4067. [p293]

E. Williamson, R. Morley, A. Lucas, and J. Carpenter. Propensity scores: From naive enthusiasm to intuitive understanding. *Statistical Methods in Medical Research*, 21(3):273–293, 2012. URL https://doi.org/10.1177/0962280210394483. [p293]

J. R. Zubizarreta. Stable weights that balance covariates for estimation with incomplete outcome data. *Journal of the American Statistical Association*, 110(511):910–922, 2015. URL https://doi.org/10.1080/01621459.2015.1023805. [p294]

Farhad Pishgar
Russell H. Morgan Department of Radiology and Radiological Science
Johns Hopkins University School of Medicine, Baltimore
United States
ORCiD: 0000-0003-0703-8442
Pishgar@JHMI.edu

Noah Greifer
Department of Mental Health
Johns Hopkins Bloomberg School of Public Health, Baltimore
United States
ORCiD: 0000-0003-3067-7154
NGreife1@JHU.edu

Clémence Leyrat
Department of Medical Statistics, Faculty of Epidemiology and Population Health
London School of Hygiene & Tropical Medicine, London
United Kingdom
ORCiD: 0000-0002-4097-4577
Clemence.Leyrat@lshtm.ac.uk

Elizabeth Stuart
Department of Mental Health & Department of Biostatistics
Johns Hopkins Bloomberg School of Public Health, Baltimore
United States
ORCiD: 0000-0002-9042-8611
EStuart@JHU.edu