An analysis of the Multivariate Imputation by Chained Equations Algorithm

Caroline Useda *
Department of Mathematics and Statistics, Amherst College

April 25, 2023

Abstract

Missing data is a common issue in statistical studies such as surveys or clinical trials, and complicates the methodology that statisticians typically use to conduct their analysis. A standard solution to this problem is to delete any observations that have missing values (known as the complete case data), but that can often result in losing a large portion of the dataset. To improve upon this solution, data imputation was created to replace missing values with predicted estimates. This paper will introduce the Multivariate Imputation by Chained Equations (MICE) algorithm by first discussing missing data in general, as well as single and multiple imputation. I will conduct an application of MICE on the iris dataset with induced missingness. This will be done by comparing MICE's results on a linear regression model with results using the full data and complete-case data.

Keywords: fully conditonal specification, sequential regression, missing data

*The author gratefully acknowledges Dr. Amy Wagaman for her help both as a professor and an advisor, as well as Eric Ingram for the idea to study MICE.

1 Introduction

In this paper, I will discuss a multiple imputation technique for missing data named multivariate imputation by chained equations (MICE). In the first section of my work, I will describe the issue of missing data as well as data imputation in general to give appropriate context for MICE. I will then examine the theory behind the method and review the steps of the algorithm. In the second section of the paper, I will conduct an application of MICE on the iris dataset by artificially deleting data. To discuss the advantages of MICE, I will execute a regression analysis on the original dataset, the MICE imputed data, and the listwise deleted data.

1.1 Missing data

Missing data is a consistent problem in statistics, especially when analyzing survey data. If participants do not fill in certain questions, it create gaps in the data that have the effect of reducing the power of a hypothesis test, creating bias in parameter estimation, and complicating the representation of the sample population. Kang (2013) There can be many different reasons for why a subject would leave a question unanswered- it could have been a complete accident, there could have been issues with visibility of the question, a question was particularly sensitive, or more. There are also examples of values that are technically missing, but the reasoning for their disappearance can be explained by the data. For example, if a respondent indicated that they don't smoke, it wouldn't make sense for them to have a value for a variable measuring years smoked. This means that not all missing data are equivalent, and there are many different types that need to be treated differently when addressing the missing values. In the next section, I will describe the types of missing data that exist before going into different methods used to prepare missing data for analysis.

1.1.1 Types of missing data

There are 3 types of missing data, first described by Rubin (1978), that classify the data based on the reason why data is missing. They are as follows Kang (2013):

1. Missing Completely at Random (MCAR): The missing values in the dataset are not related at all to the non missing values or variables in the dataset. This is rarely seen in practice, but an example would be if you purposely deleted values in a dataset for analysis.

2. Missing at Random (MAR): The missing values in the dataset are conditional on the observed data, which is independent of the missing values. Essentially, this means that the missing values in the data could be explained by the relationship between the other variables. An example of this is someone skipping a question about income, which can be estimated by the other questions on the survey.

3. Missing not at Random (MNAR): The missing values in the dataset can be easily explained, and were skipped intentionally. An example would be a participant skipping a question on drug use based on possible consequences.

1.1.2 Patterns of missing data

Another aspect that can be used to differentiate missing data is the pattern. Two concepts are important for a discussion of MICE and imputation in general:

1. Univariate/Multivariate missing data: This refers to the number of variables in the data that contain missing values. Univariate is one, while multivariate means more than one.

2. Monotone/non-monotone missing data: Missing data is said to be monotone if missing values can be ordered such that the number of missing values increases for each variable. This is something that is often seen in clinical trials, where participants can drop out as the study progresses. Any missing data that is not monotone is non-monotone.

Figure 1 from van Buuren (2018) includes a helpful visualization of the monotone missing pattern, as well as univariate. This allows us to see how univariate data could be easier to impute. While the column with missing data in the univariate pattern contains three missing values, you can see in the monotone missing data that two of the observations have

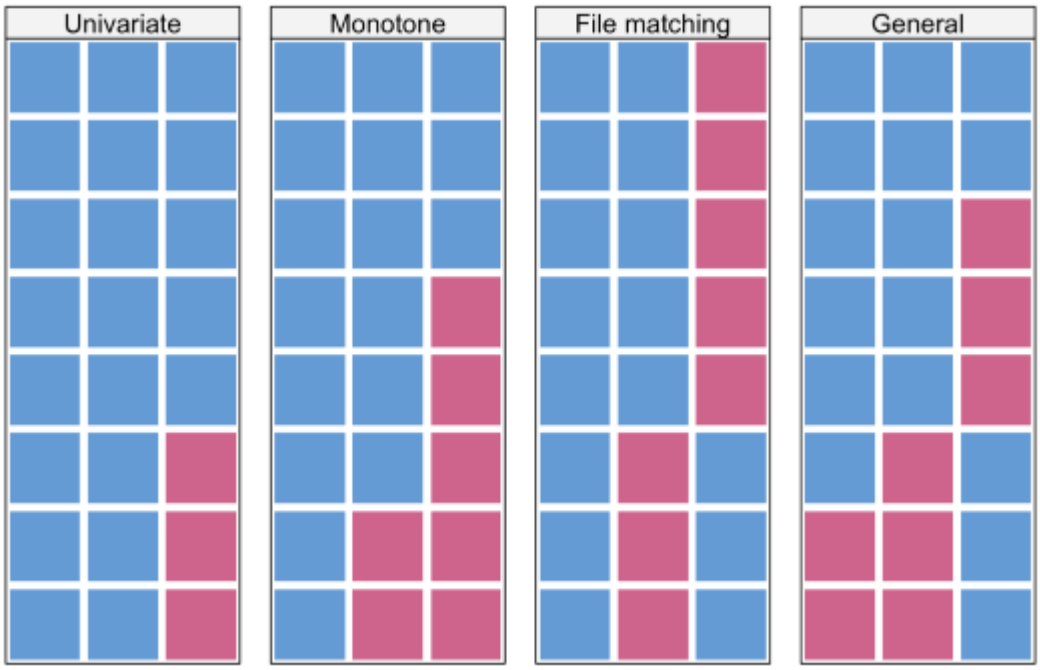

Figure 1: Visualization of data patterns

values for two out of three of their variables missing. This makes it extremely difficult to accurately predict values for those observations. General missing data is also included, where some of the observations also have two values missing, but distribution overall is spread evenly between the variables. The graphic also includes the file matching data pattern, but it is not mentioned here as it is not relevant to the introduction of MICE.

1.1.3 Data deletion

There are two common techniques for deleting missing data so that it could be used for analysis.

1. Case/listwise deletion: Any observation that has missing values is omitted. This method is also referred to as complete case analysis.

2. Pairwise deletion, where cases that have missing values are still used, just not for the variable that contains the missing value. This is also referred to as available case analysis.

A downside to both of these methods is that they rely on the data being MCAR to reduce optimal results. In other words, if there is a reason why particular data values were not included, omitting them from your analysis will bias your results. Additionally, even if your data is MCAR, if the missing values make up a large portion of the dataset, performing statistical methods on the data could prove difficult. To try and solve this issue, data imputation was developed and will be described in the following section.

1.2 Data imputation

Data imputation, or replacing missing values in a dataset with an estimate, such as the mean, has become a popular way to save a dataset that would be otherwise unusable due to the presence of missing data. There are three types of data imputation that exist currently: single, multiple, and composite. Next, I will describe single and multiple imputation as they are directly related to the MICE algorithm, but composite imputation is outside the scope of this paper. Before that discussion, I will first include a brief note on some of the issues that arise with imputing multivariate data that is relevant to all of the following methods.

1.2.1 Imputing multivariate data

I find it necessary to list a few of the common problems that arise with imputing data with multiple variables to better understand imputation in general. van Buuren (2018)

1. Variables of different types (such as categorical and numerical) make it difficult to fit a multivariate model to the data.

2. Imputed data might not always make sense, such as including the number of years smoked for someone who previously indicated that they are a nonsmoker.

3. Relationships between variables can be difficult to model using traditional statistical methods such as regression.

1.2.2 Single imputation

Single data imputation refers to the fact that only one imputed dataset is created for analysis. It does not involve defining a distribution for the dataset, and involves using one of many different techniques to estimate a missing data point. Some of the common methods are listed below:

1. Mean imputation: A missing value is replaced by the column mean.

2. Last observation carried forward: This method is applicable in situations that produce longitudinal data, such as medical trials. It replaces a missing value with the last known observation for that participant, regardless of when that occurred.

3. Regression imputation: A regression model is created using the non missing data to predict the variable(s) that contain missing. This approach is best when missing data is concentrated in a small number of columns.

Single imputation is computationally simple, but isn't always ideal. Methods such as mean imputation will replace all missing values in a column with the same value, which biases any estimate of that column. This has a larger effect of skewing results with the more missing data that is included in a dataset. To be specific, the more missing values that apply mean imputation, the closer the values for those variables get to the mean, which likely doesn't follow the true distribution of the data. To be able to accommodate this instance as well as increase the accuracy of individual estimates, Rubin proposed multiple imputation, which is discussed in the following section.

1.2.3 Multiple Imputation

First described by Rubin (1978) Rubin (1978) multiple imputation refers to the fact that this method creates multiple imputed datasets instead of just one. These datasets are then analyzed separately using traditional methods, and the results are combined to make one conclusion. Multiple imputation pools results by combining the coefficients, standard errors, and residuals from each of the m models. The algorithm can be described in 3 steps van Buuren (2018): imputation, analysis, and pooling. Figure 2 from van Buuren (2018)

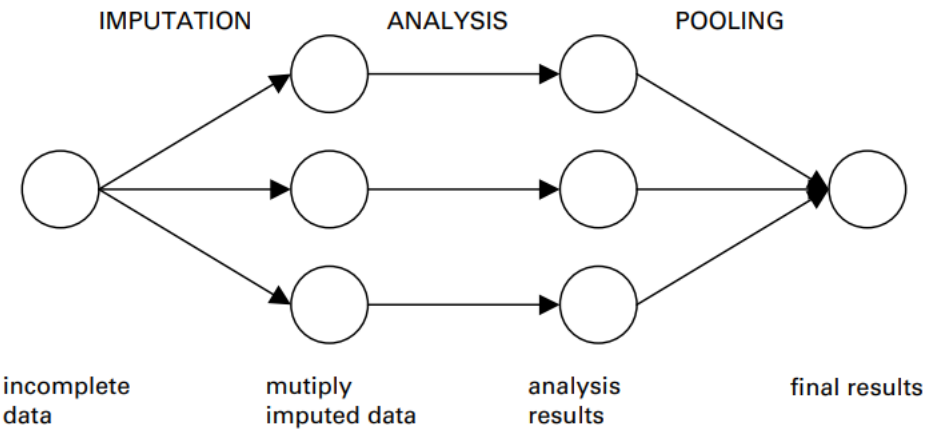

Figure 2: Outline of multiple imputation steps

is included to illustrate this process, where you can visualize how the multiple imputed datasets interact with one another to create a final result :

An example of applying multiple imputation in R to get a final pooled result is shown below, where var3 contains the missing data.

```
# creating m=5 imputed datasets
imp1 <- mice(dummydata, printFlag = FALSE, seed = 516)
# create and view pooled model
fit1 <- with(imp1, lm(var1 ~ var2 + var3))
est1 <- pool(fit1)
# summary(est1) would output a regression model using
# fitted data or, to get one of the individual datasets,
# can use complete() here, the second dataset is extracted
dummydata2 <- complete(imp1, 2)
```

One advantage of this approach is that it does not involve finding a multivariate model for the data explicitly, but rather uses the fact that it exists as justification for inference. While a vast improvement from single imputation, multiple imputation is not without it's own problems. Although it doesn't impute all missing entries in a variable with the same value, it still only uses the information present in each individual column to make it's predictions, which ignores potentially useful information when data includes important

relationships, such as those that occur when the data is MAR.

1.2.4 MICE algorithm

The issues discussed in the previous section led Van Buuren van Buuren (1999) to create the MICE algorithm, which stands for Multivariate Imputation by Chained Equations. MICE is a form of multiple imputation that works by sequentially improving regression predictions for each iteration of the algorithm. It directly incorporated mean and regression imputation by first imputing with the mean value for that column and then using those values to create a regression model to predict each missing value. This is why MICE is also referred to fully conditional specification **?** sequential regression. Instead of making the assumption in multiple imputation that you are sampling from an unspecified multivariate distribution, you instead specify a model for each variable that is conditional on the observed data. The specified model is also sequential in that each regression is linked, or rather chained to the previous iteration's model. To exemplify this, I will show the equations used when iterating through the algorithm. $\theta_p^{(t)}$ refers to the posterior distribution of variable p at iteration t, while $Y_j^{(t)}$ refers to the jth imputed variable at iteration t. Thus, there are two equations that are linked together, and they are shown below van Buuren & Groothuis-Oudshoorn (2011):

$$\theta_1^{*(t)} \sim P(\theta_1 | Y_1^{obs}, Y_2^{(t-1)}, ..., Y_p^{(t-1)})$$

$$Y_1^{*(t)} \sim P(Y_1 | Y_1^{obs}, Y_2^{(t-1)}, ..., Y_p^{(t-1)}, \theta_1^{*(t)})$$

$$\vdots$$

$$\theta_p^{*(t)} \sim P(\theta_p | Y_p^{obs}, Y_1^{(t)}, ..., Y_{p-1}^{(t)})$$

$$Y_p^{*(t)} \sim P(Y_1 | Y_p^{obs}, Y_1^{(t)}, ..., Y_p^{(t)}, \theta_p^{*(t)})$$

Next, I will outline the specific steps of the algorithm.

1. The mean value for each column is imputed for missing data values, which are placeholders.

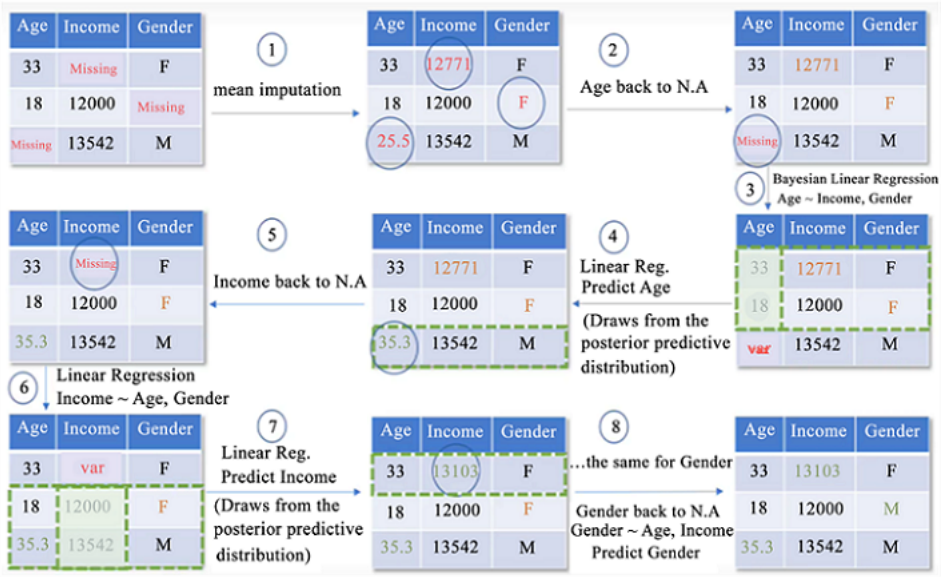

Figure 3: Diagram detailing one iteration of the MICE algorithm

2. The placeholders are deleted for the first column.

3. A regression model is created with that column as the response and all other variables as predictors. The missing values are replaced with their predictions.

4. The same is done for the next column, except you are now using the regression imputed values for all columns before the current. Continue until you reach the last column in your dataset.

5. Complete steps 2-4 until you've reached the user specified number of iterations.

An example of what a single iteration of the algorithm looks like in practice with a dataset that includes variables for age, income, and gender is featured below in Figure 3 from philip9876 (n.d.).

There are some differences in implementations of the MICE algorithm that can be adjusted. For example, step 1 can include other types of single data imputation that were discussed in the previous section. Additionally, there are different types of regression that can be used depending on the types of variables in the dataset. Next, I will demonstrate how to navigate these parameters using an application.

2 Application

For this application, I will be using the iris dataset, which contains 150 observations split evenly between 3 species of flowers. The other variables included in the dataset are sepal.length, petal.length, sepal.width, and petal.width. To perform Multivariate Imputation by Chained Equations on the dataset that doesn't naturally have missing data, I will be randomly deleting values from one variable, sepal.length, such that the resulting dataset is MAR. I am able to do so by assigning different probabilities for the observation being missing based on it's value of sepal.length: If sepal.length is below about the midpoint, it will have a lower probability of being missing, and a higher probability if above the midpoint. I will then create a complete case dataset as well as a imputed dataset. To compare the three datasets to each other, I will perform regression analysis on the full dataset, the complete case dataset, and the MICE data.

```
# loading original dataset
data(iris)
iris$Species <- as.factor(iris$Species)

# checking range of sepal.length
range(iris$Sepal.Length)
```

```
## [1] 4.3 7.9
```

```
# plot of sepal.length values
ggplot(iris, mapping = aes(Sepal.Length)) + geom_histogram(fill = "dark blue",
    bins = 35) + labs(caption = "Figure 4: Distribution of sepal.length") +
    xlab("sepal.length (cm)") + theme(plot.caption = element_text(hjust = 0.5))
```

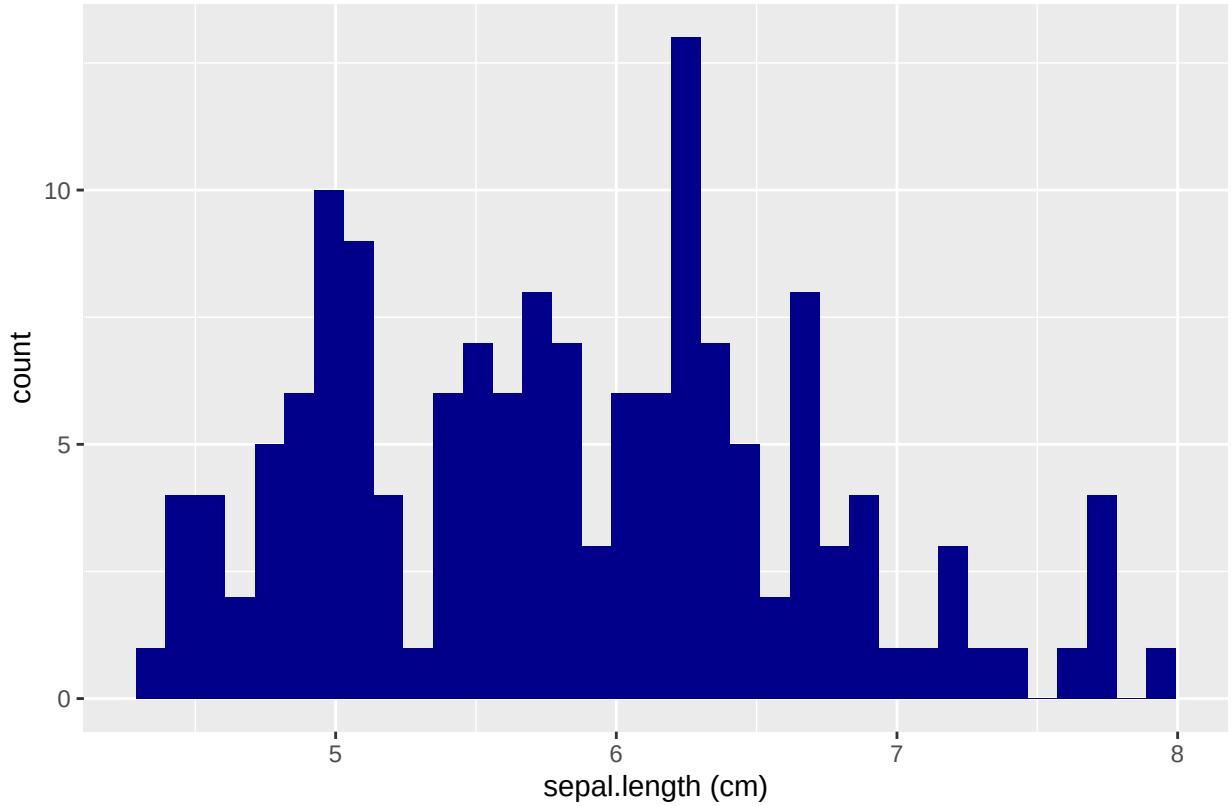

Figure 4: Distribution of sepal.length

```
# relationships between variables
cor(iris$Sepal.Length, iris$Sepal.Width)
```

```
## [1] -0.1175698
```

```
cor(iris$Sepal.Length, iris$Petal.Length)
```

```
## [1] 0.8717538
```

```
cor(iris$Sepal.Length, iris$Petal.Width)
```

```
## [1] 0.8179411
```

```r
# selecting which values to delete
set.seed(516)
iris3 <- mutate(iris, missing = 0)
for (i in 1:nrow(iris3)) {
    test1 <- sample(0:1, 1, prob = c(0.4, 0.6))
    test2 <- sample(0:1, 1, prob = c(0.8, 0.2))
    missingval <- ifelse(iris3$Sepal.Length[i] > 5.5, test1,
        test2)
    iris3$missing[i] <- missingval
}

# creating dataset with missing values
iris4 <- iris3 %>%
    mutate(Sepal.Length = ifelse(missing == 1, NA, Sepal.Length))

# deleting indicator variable 'missing' this is not good
# coding practice! not reproducible because refers to a
# specific number and only works here as 6 is last column
iris4 <- iris4[, -6]

# creating complete case dataset
complete <- iris4 %>%
    filter(Sepal.Length != is.na(Sepal.Length))

# seeing dist of species in complete case data
count(complete, Species)
```

```
##      Species  n
## 1     setosa 42
## 2 versicolor 24
## 3  virginica 21
```

Here, you can see that sepal.length values are between 4.3cm and 7.9cm and are pretty uniformly distributed, with some skew toward smaller values. With that being said, I split the data at sepal.length = 5.5 cm, with values below having a 40% probability of being missing, and values above 80%. This lead to 63 values being deleted in the dataset intended for MICE imputation, with the complete case dataset having 87 observations. Additionally, there seems to be significantly less observations missing from the setosa species than the other two. Lastly, there do appear to be significant relationships present in the data, with sepal.length having correlations above .8 with both petal.length and petal.width. Next, we will apply MICE and examine the imputed values.

```
# mice imputation
miceimp <- mice(iris4, seed = 516, printFlag = FALSE)

# plot of imputed vs. real values
ggmice::ggmice(miceimp, aes(x = .imp, y = Sepal.Length)) + geom_point() +
    labs(caption = "Figure 5: Imputed values vs. Observed values") +
    theme(plot.caption = element_text(hjust = 0.5))
```

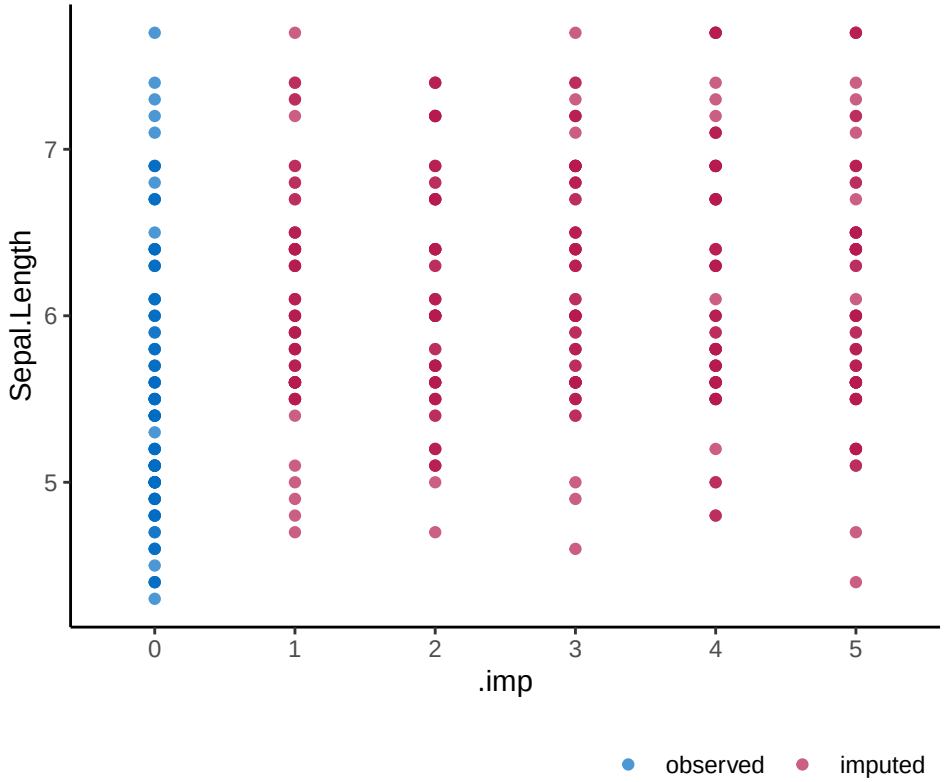

Figure 5: Imputed values vs. Observed values

```
# comparing to missing values
range(complete$Sepal.Length)
```

```
## [1] 4.3 7.7
```

```
# looking how imputed values changed per iteration
ggmice::plot_trace(miceimp, "Sepal.Length") + labs(caption = "Figure 6: Imputed values
       for each iteration") +
    theme(plot.caption = element_text(hjust = 0.5))
```

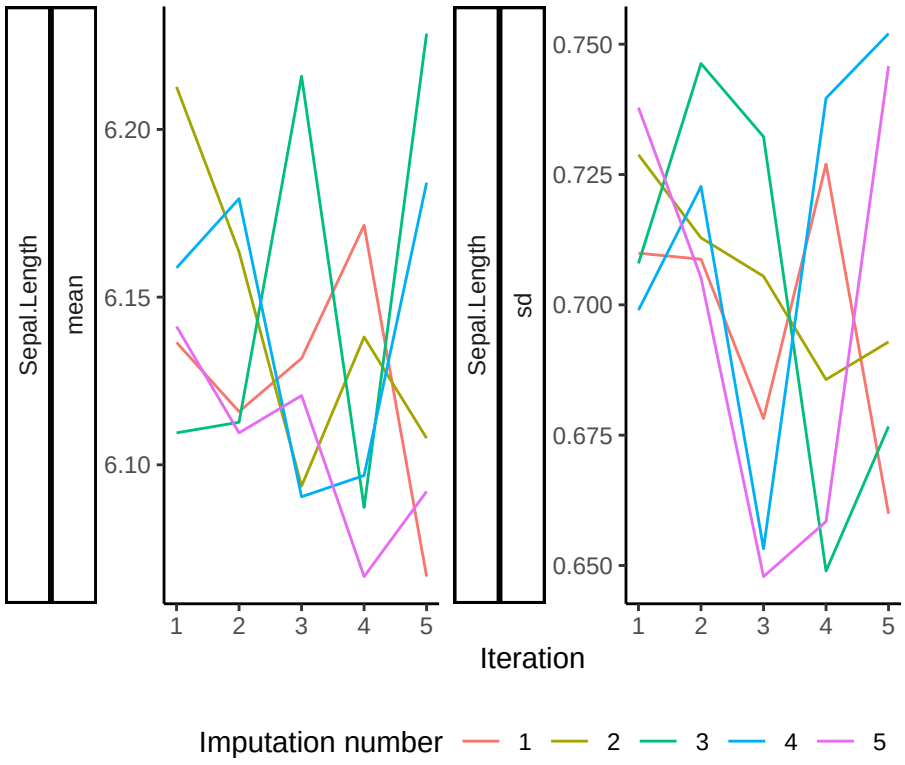

Figure 6: Imputed values for each iteration

These imputations seems reasonable because they are very similar to the values that already existed for sepal.length. Figure 5 shows that the overall distribution of the variable is similar, with pink dots representing the imputed values. The most notable differences is that MICE seemed to put more observations at the upper range of sepal.length than the original data, and less observations at the lower range. However, comparing this with the range of values in the complete case dataset, which is 4.3 cm to 7.7 cm, we can see that this makes sense: there are more missing observations from higher sepal.length values. We also know this since sepal.length values over 5.5 cm had a higher probability of being missing. Thus, Figure 5 shows us how the imputed datasets come together to fill the gap made by the missing data. I also looked at how the imputed values for each of the five datasets changed for each of the five iterations of the algorithm in Figure 6. Both the mean and standard deviation have two datasets that have higher values, and the rest have lower values. Additionally, looking at the variation within a single dataset allows us to understand how variable imputations can be from iteration to iteration. This shows the importance

of pooling results from different datasets, as well as the variance in imputed values even within a single dataset. Next, I will run a regression on all three of the datasets that have been creates: full, complete case, and MICE imputed data. I will predict petal.length using the remaining variables, noting that the range for this variable is between 1 cm and 6.9 cm.

```
# real regression
fullreg <- lm(Petal.Length ~ Sepal.Length + Sepal.Width + Petal.Width +
    Species, data = iris)
summary(fullreg)
```

```
##
## Call:
## lm(formula = Petal.Length ~ Sepal.Length + Sepal.Width + Petal.Width +
##     Species, data = iris)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.78396 -0.15708  0.00193  0.14730  0.65418
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -1.11099    0.26987  -4.117 6.45e-05 ***
## Sepal.Length       0.60801    0.05024  12.101  < 2e-16 ***
## Sepal.Width       -0.18052    0.08036  -2.246   0.0262 *
## Petal.Width        0.60222    0.12144   4.959 1.97e-06 ***
## Speciesversicolor  1.46337    0.17345   8.437 3.14e-14 ***
## Speciesvirginica   1.97422    0.24480   8.065 2.60e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.2627 on 144 degrees of freedom
## Multiple R-squared:  0.9786, Adjusted R-squared:  0.9778
## F-statistic:  1317 on 5 and 144 DF,  p-value: < 2.2e-16
```

```
# complete regression
completereg <- lm(Petal.Length ~ Sepal.Length + Sepal.Width +
    Petal.Width + Species, data = complete)
summary(completereg)
```

```
##
## Call:
## lm(formula = Petal.Length ~ Sepal.Length + Sepal.Width + Petal.Width +
##     Species, data = complete)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.74853 -0.12732  0.02573  0.13781  0.49063
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -0.88641    0.31655  -2.800  0.00638 **
## Sepal.Length       0.60336    0.06548   9.214 2.99e-14 ***
## Sepal.Width       -0.24648    0.09101  -2.708  0.00825 **
## Petal.Width        0.65959    0.14682   4.492 2.31e-05 ***
## Speciesversicolor  1.35679    0.19550   6.940 8.80e-10 ***
## Speciesvirginica   1.81882    0.27762   6.551 4.88e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2259 on 81 degrees of freedom
## Multiple R-squared:  0.9848, Adjusted R-squared:  0.9838
```

```
## F-statistic:  1049 on 5 and 81 DF,  p-value: < 2.2e-16
```

```
# mice regression
set.seed(516)
fit <- with(miceimp, lm(Petal.Length ~ Sepal.Length + Sepal.Width +
    Petal.Width + Species))
est <- pool(fit)
estdata <- est$pooled %>%
    select(term, estimate, t, df)
kable(estdata)
```

term	estimate	t	df
(Intercept)	-1.2582567	0.0784169	77.43557
Sepal.Length	0.6664938	0.0026797	123.32245
Sepal.Width	-0.2280019	0.0062072	133.34616
Petal.Width	0.6268393	0.0217729	20.42043
Speciesversicolor	1.4103973	0.0331839	62.84755
Speciesvirginica	1.8421372	0.0814015	27.70172

Looking at the coefficients for all three models, it appears that both the complete case and MICE imputed data have similar regression coefficients to the full data. To compare the models more easily, let's look at a table of the differences in the coefficients when comparing to the full dataset. Specifically, I will be subtracting the complete case and MICE coefficients from the full regression coefficients, as well as getting the average differences.

```
# getting coefficients from respective models
fullcoef <- fullreg$coefficients
completecoef <- completereg$coefficients
micecoef <- summary(est)$estimate
# calculating the differences
completediff <- abs(fullcoef - completecoef)
micediff <- abs(fullcoef - micecoef)
```

```
# getting averages
mean(completediff)
```

```
## [1] 0.1024239
```

```
mean(micediff)
```

```
## [1] 0.07715188
```

```
mean(completediff) - mean(micediff)
```

```
## [1] 0.02527201
```

```
# creating table
diffdata <- data.frame(completediff, micediff) %>%
    rename(`Complete Diff` = completediff, `MICE Diff` = micediff)
kable(diffdata)
```

	Complete Diff	MICE Diff
(Intercept)	0.2245762	0.1472680
Sepal.Length	0.0046504	0.0584880
Sepal.Width	0.0659571	0.0474784
Petal.Width	0.0573696	0.0246178
Speciesversicolor	0.1065856	0.0529736
Speciesvirginica	0.1554043	0.1320856

Here, it becomes clear that the MICE model was closer to the full dataset, with a average difference in coefficients being .025 units lower than that of the complete case model. Looking at the differences for each individual variable, larger differences are seen in the intercept as well as the factor variable for Speciesvirginicia. Since sepal.length was the imputed variable, it is important to note that in this instance, the complete case model came significantly closer to the full model's coefficient. One reason for this performance could lie in the fact that complete case linear regression models are often unbiased under the MAR assumption.

2.1 Conclusion

In this paper, I introduced the Multiple Imputation by Chained Equations (MICE) algorithm by discussing missing data and data imputation in general. I have found that MICE is a useful multiple imputation method for imputing multivariate missing data, but has some drawbacks such as including a strong assumption that the data is Missing at Random (MAR). Through my application of MICE onto the iris dataset, I showed how to perform and analyze MICE results as well as compare those results to the full dataset with no missing data and complete case data using linear regression. I found that while both methods for handling missing data performed well, the MICE model found coefficients more similar to those observed in the full model.

2.1.1 Limitations

I want to acknowledge some limitations to the results provided in the application. MICE was only applied once to a dataset with 150 observations, so my analysis cannot be extended to other types of data settings. I also only removed observations from one variable, and including others could have changed the accuracy of the complete case model. Finally, I ran the MICE algorithm with all of the standard parameters, and adjusting them could have led the MICE model to perform better.

2.2 Further work

There is a lot of work that can be done to further research this subject. Including a simulation where you generate the data yourself would allow for more comparisons to be made, as well as more repetitions that makes the results more reliable. MICE also could have been compared to other methods than complete case, such as any of the single imputation methods that were discussed or other multiple imputation algorithms. Lastly, it could be interesting to see comparisons between methods other than linear regression models.

References

Kang, H. (2013), 'The prevention and handling of the missing data', *Korean Journal of Anesthesiology* **64**(5), 402–406.

philip9876 (n.d.), 'Multiple imputation by chained equations (mice)'.
URL: *https://www.youtube.com/watch?v=zX-pacwVyvU*

Rubin, D. B. (1978), '"multiple imputations in sample surveys—a phenomenological bayesian approach to nonresponse"', *Proceedings of the Survey Research Methods Section of the American Statistical Association* pp. 20–28.

van Buuren, S. (1999), 'Flexible multivariate imputation by mice', *TNO prevention and Health* .

van Buuren, S. (2018), *Flexible Imputation of Missing Data, Second Edition (2nd ed.)*, Chapman and Hall/CRC.
URL: *https://doi.org/10.1201/9780429492259*

van Buuren, S. & Groothuis-Oudshoorn, C. (2011), 'Mice: Multivariate imputation by chained equations in r', *Journal of Statistical Software* .