Handling Missing Data in R Workshop

Delivered by: Rahul Patel, Sally Xie

Table of contents

1 Before the Workshop

1.1 Download R studio

Before participating in the workshop please download RStudio (this will also require you to download R if you have not used this software before). RStudio uses the same syntax and functions as R, but has a more user-friendly interface.

- When using RStudio we want to make sure that all of our files are in the same place. Therefore, you should create a separate folder somewhere on your computer and save all the files provided there.

1.2 Download Quarto

Quarto is a multi-language, next generation version of R Markdown from Posit, with many new features and capabilities. Like R Markdown, Quarto uses knitr to execute R code, and is therefore able to render most existing Rmd files without modification. To download Quarto, visit https://quarto.org/docs/get-started/ and click on the download link that corresponds with your operating system.

1.3 Download Latex

LaTeX is not a stand-alone typesetting program in itself, but document preparation software that runs on top of Donald E. Knuth's TeX typesetting system. TeX distributions usually bundle together all the parts needed for a working TeX system and they generally add to this both configuration and maintenance utilities. Nowadays LaTeX, and many of the packages built on it, form an important component of any major TeX distribution. Quarto leverages LaTeX, so it is important to download it as well. Visit https://www.latex-project.org/get/.

2 Introduction

This workbook will guide you through the complexities of managing missing data in R, covering foundational concepts, visualizing missing data, and the practical application of multiple imputation using a dataset.

3 Packages

We will install the following packages:

tidyverse The `tidyverse` is a collection of R packages designed for data science that share an underlying design philosophy, grammar, and data structures. The suite includes packages like `ggplot2` for data visualization, `dplyr` for data manipulation, `tidyr` for tidying data, among others. It is widely used for its coherent syntax and functionality that makes data manipulation, exploration, and visualization easier and more intuitive.

naniar The `naniar` package provides structured methods for handling missing values in data. It extends the tidyverse style of data manipulation to missing data, providing easy-to-use functions and visualizations to diagnose and manage missingness in datasets. This package helps in making the exploration and treatment of missing data seamless and integrated within the tidyverse workflow.

VIM `VIM` (Visual Impairment Methods) offers powerful visualization tools to uncover missing data patterns and structures. It helps in the imputation of missing values through various plots, such as missing value heat maps, scatter plots, and diagnostic plots, which can inform the imputation strategy and improve the understanding of the nature of the missing data.

mice The `mice` package (Multivariate Imputation by Chained Equations) provides a framework for dealing with missing data in a principled manner. It implements multiple imputation using chained equations, allowing for the imputation of missing values in a way that is statistically sound and flexible, supporting various types of data and imputation models.

ggmice `ggmice` is an extension of the `mice` package and integrates with `ggplot2` from the tidyverse to visualize imputed data. It is designed to create plots from multiple imputed datasets easily, providing a convenient way to assess the variability of the imputations and to communicate the results of the multiple imputation process.

diffdf `diffdf` is an R package used to compare two data frames and identify differences. It is particularly useful in data cleaning and validation phases of data analysis, allowing users to assert that two datasets are the same or to pinpoint exactly where and how they differ. This package supports detailed comparisons including type checks, key variable checks, and provides comprehensive reports on the discrepancies found.

After opening RStudio, in the console type the following code to install the necessary packages

Make sure you are connected to the internet when installing the packages.

```r
install.packages("tidyverse")
install.packages("naniar")
install.packages("VIM")
install.packages("mice")
install.packages("ggmice")
install.packages("diffdf")
```

```r
library(tidyverse)
library(naniar)
library(VIM)
library(mice)
library(ggmice)
library(diffdf)
```

4 Read and Load the Data

Read the missing_perceptions_jsat.csv data file. This data file has simulated missing values. This data file has 5 variables: `gender`, `age`, `race`, `edu` (education), and `jsat` (job satisfaction). The scale for `jsat` uses a 7-point Likert type scale, and ranges from strongly disagree to strongly agree.

```
# Load the data using the read_csv() command
missing_perceptions_jsat_data <- read_csv("missing_perceptions_jsat_data.csv")
```

Examines the structure of the missing data

```
# Examine the structure of the data: variables and data entries
glimpse(missing_perceptions_jsat_data)
```

```
Rows: 268
Columns: 5
$ gender <chr> "Male", "Gender Minority", "Gender Minority", "Male", "Gender M~
$ age    <dbl> 28, 24, 28, 29, 25, 25, 50, 28, 72, 29, NA, 24, 37, 29, 59, 23,~
$ race   <chr> NA, "South Asian", "East Asian", "White", "White", "White", NA,~
$ edu    <chr> "Completed College/University", "Completed College/University",~
$ jsat   <dbl> 4.6, NA, 4.8, 6.1, NA, 5.6, 7.0, 4.7, 5.9, 5.9, 5.7, 5.7, 5.8, ~
```

> **Note**
>
> Notice how gender, race, and edu are `<dbl>` type variables. We need to make these `<fct>` (factor) type variables otherwise they will be imputed differently and incorrectly.

Before we get there, the NA values are stored as `<chr>`, and this creates problems for when we convert `gender`, `race`, and `edu` to factors. Below is code to circumvent this issue.

```
missing_perceptions_jsat_data <- missing_perceptions_jsat_data %>%
  mutate(gender = factor(gender,
                         levels = unique(missing_perceptions_jsat_data$gender)),
         race = factor(race,
                       levels = unique(missing_perceptions_jsat_data$race)),
         edu = factor(edu,
                      levels = unique(missing_perceptions_jsat_data$edu))) %>%
  as.data.frame()

# Again, examine the structure of the data to confirm conversions
```

```r
glimpse(missing_perceptions_jsat_data)
```

```
Rows: 268
Columns: 5
$ gender <fct> Male, Gender Minority, Gender Minority, Male, Gender Minority, ~
$ age    <dbl> 28, 24, 28, 29, 25, 25, 50, 28, 72, 29, NA, 24, 37, 29, 59, 23,~
$ race   <fct> NA, South Asian, East Asian, White, White, White, NA, White, Wh~
$ edu    <fct> Completed College/University, Completed College/University, Com~
$ jsat   <dbl> 4.6, NA, 4.8, 6.1, NA, 5.6, 7.0, 4.7, 5.9, 5.9, 5.7, 5.7, 5.8, ~
```

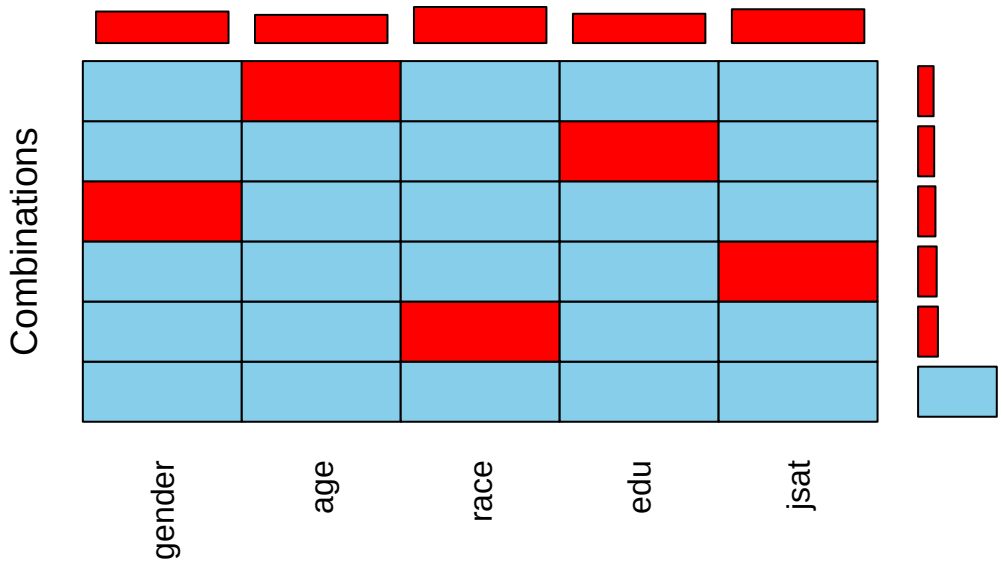
5 Visualize Missing Data

5.1 Dataset-Level

To visualize data at the dataset-level, we create **aggregation plots**. Aggregation plots tell us in which variables the data are missing and how often.

- The top red bars represent the proportion of missing data for each variable
- The red bars on the right represent the proportion of missing data, broken down by combinations of missingness. For instance, looking at the first row, it appears that 9% of observations have missing data where age is missing. Overall, it appears no observations have missing values where there are more than 2 variables missing. The blue bar in the bottom right represents the proportion of observed data that has no missing data.

```
missing_perceptions_jsat_data %>%
  aggr(combine = TRUE, numbers = TRUE)
```



5.2 Variable-Level

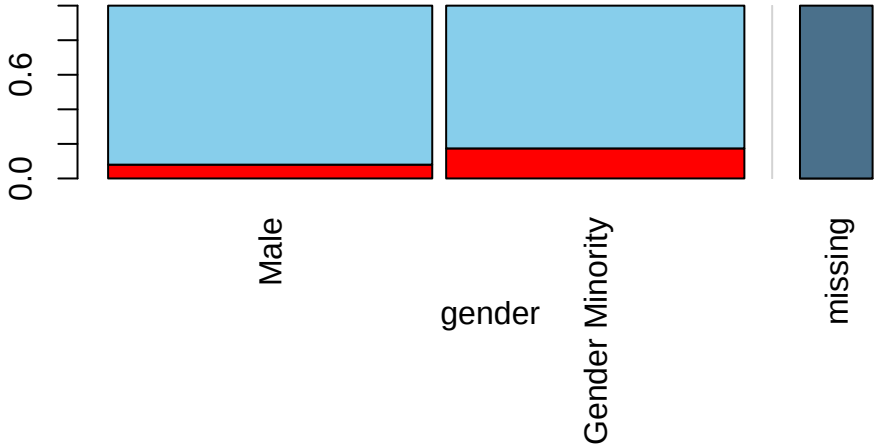
At the variable-level, we can create **spine plots**. Spine plots allows us to study the percentage of missing values in one variable for different values of the other.

- The width of bars represent the proportion of those values that are present in the data.
- The red bars represent the proportion of missing data.

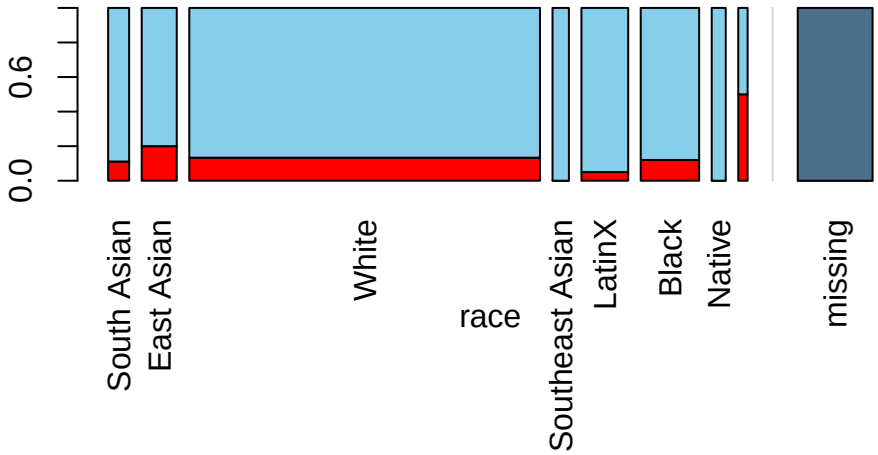
- Blue bars represent the proportion of observed data.

Below are spine plots depicting missing values in jsat by gender and jsat by race.

```
# Set margins (bottom, left, top, right)
par(mar = c(10, 3, 3, 3))
# Draw a spine plot to analyse missing values in jsat by gender
missing_perceptions_jsat_data %>%
  select(gender, jsat) %>%
  spineMiss()
```

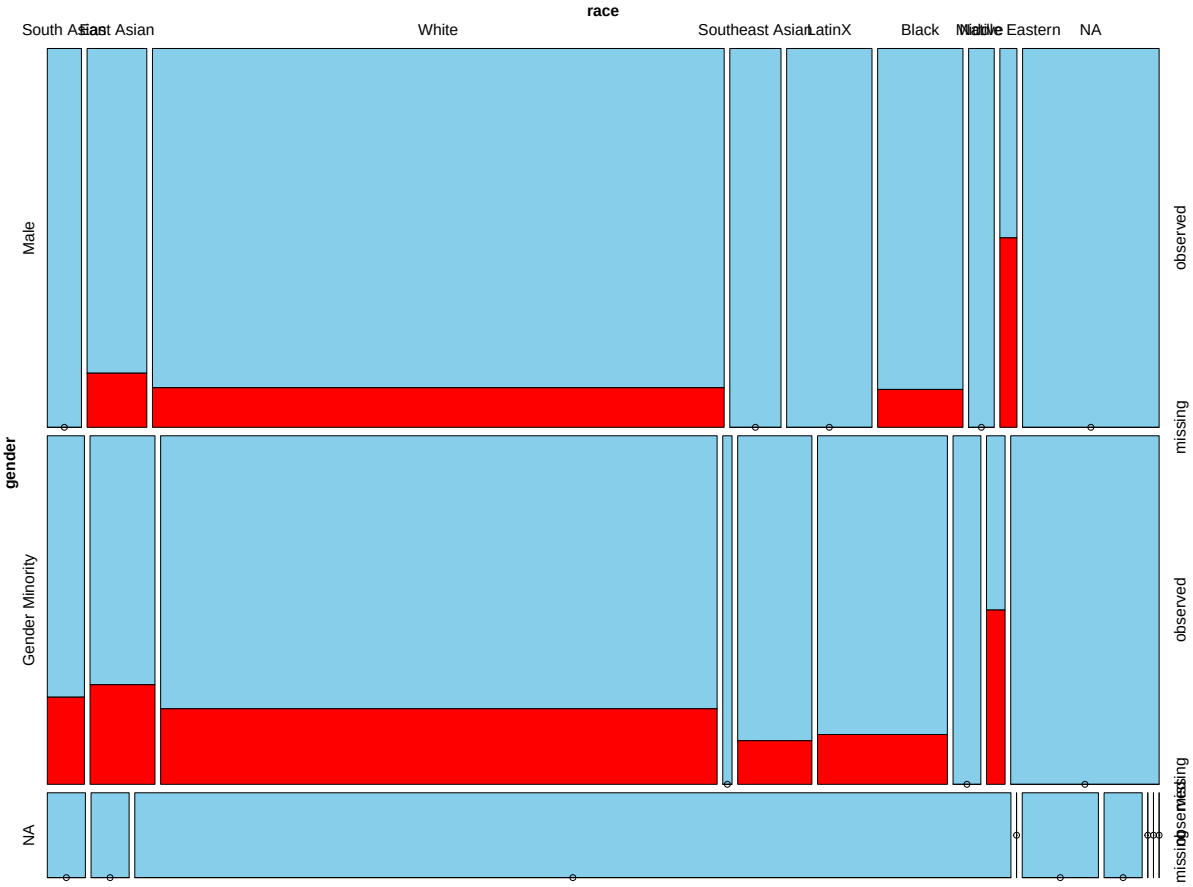



```
# Draw a spine plot to analyse missing values in jsat by race
missing_perceptions_jsat_data %>%
  select(race, jsat) %>%
  spineMiss()
```



We can also create **mosaic plots**. Mosaic plots can be thought of as a generalization of the spine plot to more variables. Specifically, we can look at missing data for a continuous variable, broken down by more than 1 categorical variable. This plot is a collection of tiles, where each tile corresponds to a specific combination of categories (for categorical variables) or bins (for numeric variables).

```
missing_perceptions_jsat_data %>%
  mosaicMiss(highlight = "jsat",
             plotvars = c("gender", "race"))
```



6 Multiple Imputation with MICE

MICE, or Multivariate Imputation by Chained Equations, is a statistical method used to handle missing data in multivariate datasets. It operates by performing multiple imputations which predict missing values using the observed data, thereby addressing the problem of incomplete datasets in a robust and statistically sound manner.

Journal of Statistical Software

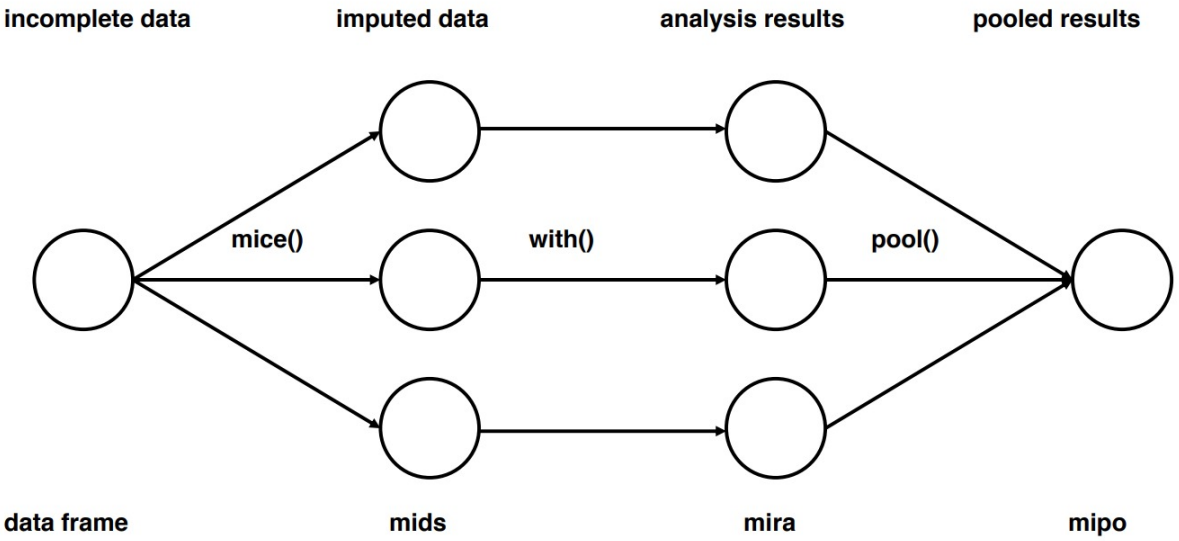


Figure 1: Main steps used in multiple imputation.

Below we impute the missing data with `mice()`

```
missing_perceptions_jsat_data_multiimp <- mice(missing_perceptions_jsat_data,
                                               m = 5, # ideally, specify based on
                                               # proportion of missingness (e.g., if 0.5,
                                               # specify 50)
                                               method = c("logreg", "pmm", "polyreg",
                                                          "polr", "pmm"),
                                               seed = 1234,
                                               maxit = 5) # ideally, specify between 20-30
```

```
 iter imp variable
  1   1  gender  age  race  edu  jsat
  1   2  gender  age  race  edu  jsat
  1   3  gender  age  race  edu  jsat
  1   4  gender  age  race  edu  jsat
  1   5  gender  age  race  edu  jsat
  2   1  gender  age  race  edu  jsat
  2   2  gender  age  race  edu  jsat
  2   3  gender  age  race  edu  jsat
  2   4  gender  age  race  edu  jsat
  2   5  gender  age  race  edu  jsat
  3   1  gender  age  race  edu  jsat
  3   2  gender  age  race  edu  jsat
  3   3  gender  age  race  edu  jsat
  3   4  gender  age  race  edu  jsat
  3   5  gender  age  race  edu  jsat
  4   1  gender  age  race  edu  jsat
  4   2  gender  age  race  edu  jsat
  4   3  gender  age  race  edu  jsat
  4   4  gender  age  race  edu  jsat
  4   5  gender  age  race  edu  jsat
  5   1  gender  age  race  edu  jsat
  5   2  gender  age  race  edu  jsat
  5   3  gender  age  race  edu  jsat
  5   4  gender  age  race  edu  jsat
  5   5  gender  age  race  edu  jsat
```

```
print(missing_perceptions_jsat_data_multiimp)
```

```
Class: mids
Number of multiple imputations:  5
Imputation methods:
   gender       age      race       edu      jsat
 "logreg"     "pmm" "polyreg"    "polr"     "pmm"
PredictorMatrix:
       gender age race edu jsat
gender      0   1    1   1    1
age         1   0    1   1    1
race        1   1    0   1    1
edu         1   1    1   0    1
jsat        1   1    1   1    0
```

```
Number of logged events:  21
  it im  dep    meth                 out
1  1  1 race polyreg eduSome High School
2  1  2 race polyreg eduSome High School
3  1  3 race polyreg eduSome High School
4  1  4 race polyreg eduSome High School
5  1  5 race polyreg eduSome High School
6  2  1 race polyreg eduSome High School
```

Let's break down the arguments in the `mice()` function:

- **m:** Number of multiple imputations. The default is `m=5`. Ideally, you specify this based on the proportion of missingness in your data (e.g., if 0.5, specify 50).
- **method:**
  - Can be either a single string, or a vector of strings with length `length(blocks)`, specifying the imputation method to be used for each column in data. If specified as a single string, the same method will be used for all blocks. The default imputation method (when no argument is specified) depends on the measurement level of the target column, as regulated by the `defaultMethod` argument. Columns that need not be imputed have the empty method `""`.
  - In our case, `gender` is an unordered categorical variable with 2 levels and imputations of these variables with `logreg` is appropriate; `race` is similarly an unordered categorical variable, but with more than 2 levels, and imputations of these variables with `polyreg` is appropriate; `age` is numeric, along with `jsat`, and numeric variables are most appropriately imputed with `pmm`; `education` is an ordered categorical variable, and variables of this type are best handled with `polr.`
  - `logreg`: Logistic Regression (logreg) is used for imputing binary categorical data. It models the probability that an outcome belongs to a particular category (typically 0 or 1). In the context of MICE, logistic regression is used to predict missing binary values based on the observed portions of the data. The imputed values are drawn from a Bernoulli distribution, where the probability of occurrence of one category (e.g., 1) is predicted by the logistic model.
  - `pmm`: Predictive Mean Matching (PMM) is a semi-parametric imputation method that is less sensitive to outliers than purely parametric methods like normal regression. PMM works by selecting a donor pool from the observed values whose predicted values are closest to the predicted value of the missing point. An observed value is then randomly selected from this donor pool to replace the missing value.
  - `polyreg`: Polynomial Regression (polyreg) is used for imputing categorical variables with more than two categories. It employs a multinomial logistic regression model,

treating each category as a possible outcome and modeling the probabilities of each category based on the predictors. The missing values are then imputed based on the probabilities estimated for each category.

  - `polr`: Proportional Odds Model (polr) is used for ordinal data, where the categories have a natural ordering but intervals between categories are not assumed to be equal (e.g., satisfaction ratings like low, medium, high). It works under the assumption that the relationship between each pair of outcome groups is statistically similar. Thus, instead of modeling the probability for each category separately, it models the cumulative probability up to a certain category.

- **seed:** An integer that is used as argument by the `set.seed()` for offsetting the random number generator. Specify this so that others can have the same results as you—for reproducibility!
- **maxit:** A scalar giving the number of iterations. The default is 5. Ideally, specify between 20-30 as more iterations tend to be better.
- **predictorMatrix:** We didn't specify this matrix. In MICE, all variables are imputed by all other variables. predictorMatrix allows you to specify which variables are imputing which variables, which could be useful depending on your domain or theory.

You can view the imputations using the above imputed data set

```
# View all imputations
missing_perceptions_jsat_data_multiimp$imp

# View imputations by variable
missing_perceptions_jsat_data_multiimp$imp$jsat
```

You can also obtain the imputed dataset, or even one of the iterations using `complete()`

```
# Obtain the imputed dataset
imputed_dataset_exported <- complete(missing_perceptions_jsat_data_multiimp)

# Or, can obtain one of the imputed datasets. Note, we specified 5 imputations,
# so there are 5 imputed datasets to pull from; we can specify from 1 all the way to 5
print(head(complete(missing_perceptions_jsat_data_multiimp, 1), n = 25))
```

```
            gender age           race                                edu jsat
1             Male  28     East Asian    Completed College/University  4.6
2  Gender Minority  24     South Asian   Completed College/University  7.0
3  Gender Minority  28     East Asian    Completed College/University  4.8
4             Male  29          White    Completed College/University  6.1
```

```
5  Gender Minority  25           White     Completed College/University  5.7
6              Male  25           White     Completed College/University  5.6
7  Gender Minority  50           Black Completed Postgraduate Education  7.0
8              Male  28           White     Completed College/University  4.7
9  Gender Minority  72           White            Apprenticeship/Trades  5.9
10             Male  29           White     Completed College/University  5.9
11             Male  52 Southeast Asian          Some College/University  5.7
12             Male  24          LatinX          Some College/University  5.7
13 Gender Minority  37           White     Completed College/University  5.8
14             Male  29           White     Completed College/University  4.4
15 Gender Minority  59           White          Some College/University  6.0
16 Gender Minority  23           White     Completed College/University  4.2
17 Gender Minority  34           White     Completed College/University  6.6
18             Male  40           White Completed Postgraduate Education  6.6
19 Gender Minority  25           White     Completed College/University  3.6
20             Male  53           White     Completed College/University  5.3
21 Gender Minority  67           Black          Some College/University  4.8
22 Gender Minority  46           White          Some College/University  4.3
23 Gender Minority  21           White     Completed College/University  4.4
24             Male  34           White Completed Postgraduate Education  5.9
25 Gender Minority  51           White Completed Postgraduate Education  5.6
```

```
# Compare differences in the original versus imputed dataset
diffdf(missing_perceptions_jsat_data, imputed_dataset_exported)
```

```
Differences found between the objects!

A summary is given below.

Not all Values Compared Equal
All rows are shown in table below

  ============================
   Variable  No of Differences
  ----------------------------
    gender          28
     age            25
     race           32
     edu            26
     jsat           30
  ----------------------------
```

First 10 of 28 rows are shown in table below

VARIABLE	..ROWNUMBER..	BASE	COMPARE
gender	6	<NA>	Male
gender	9	<NA>	Gender Minority
gender	12	<NA>	Male
gender	15	<NA>	Gender Minority
gender	26	<NA>	Gender Minority
gender	32	<NA>	Gender Minority
gender	42	<NA>	Gender Minority
gender	44	<NA>	Gender Minority
gender	53	<NA>	Male
gender	66	<NA>	Gender Minority

First 10 of 25 rows are shown in table below

VARIABLE	..ROWNUMBER..	BASE	COMPARE
age	11	<NA>	52
age	23	<NA>	21
age	30	<NA>	40
age	46	<NA>	60
age	48	<NA>	72
age	57	<NA>	35
age	61	<NA>	60
age	63	<NA>	48
age	64	<NA>	43
age	85	<NA>	26

First 10 of 32 rows are shown in table below

VARIABLE	..ROWNUMBER..	BASE	COMPARE

race	1	<NA>	East Asian
race	7	<NA>	Black
race	25	<NA>	White
race	36	<NA>	White
race	45	<NA>	White
race	60	<NA>	Black
race	89	<NA>	Black
race	93	<NA>	White
race	104	<NA>	White
race	107	<NA>	White

First 10 of 26 rows are shown in table below

VARIABLE	..ROWNUMBER..	BASE	COMPARE
edu	13	<NA>	Completed College/University
edu	14	<NA>	Completed College/University
edu	18	<NA>	Completed Postgraduate Educati...
edu	20	<NA>	Completed College/University
edu	28	<NA>	Completed Postgraduate Educati...
edu	35	<NA>	Completed High School
edu	39	<NA>	Some College/University
edu	50	<NA>	Completed High School
edu	58	<NA>	Completed Postgraduate Educati...
edu	69	<NA>	Completed Postgraduate Educati...

First 10 of 30 rows are shown in table below

VARIABLE	..ROWNUMBER..	BASE	COMPARE
jsat	2	<NA>	7.0
jsat	5	<NA>	5.7
jsat	17	<NA>	6.6
jsat	22	<NA>	4.3
jsat	34	<NA>	5.9
jsat	51	<NA>	7.0
jsat	54	<NA>	6.2

```
   jsat         56        <NA>    5.0
   jsat         59        <NA>    7.0
   jsat         67        <NA>    5.1
----------------------------------------
```

7 Assessing Imputation Quality: Visualization

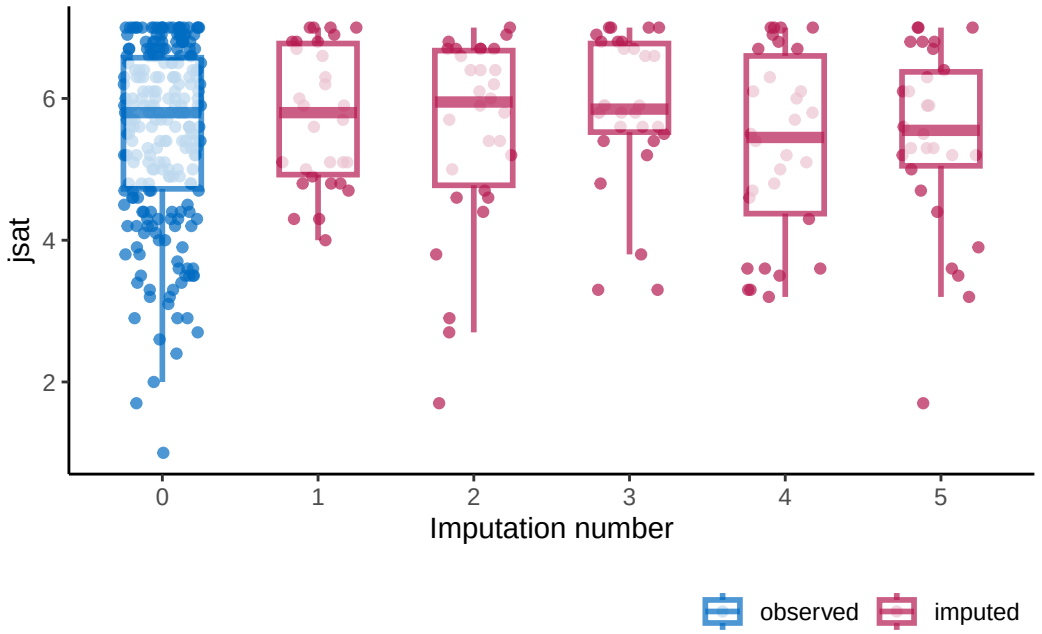
7.1 Variable-level

There are different ways of going about assessing imputation quality at the variable level. At the univariate level, we can compare observed data with imputations using **scatterplots** and **density plots**.

> **!** Important
>
> Ideally, the pattern of imputed data points should match the pattern of observed data points.

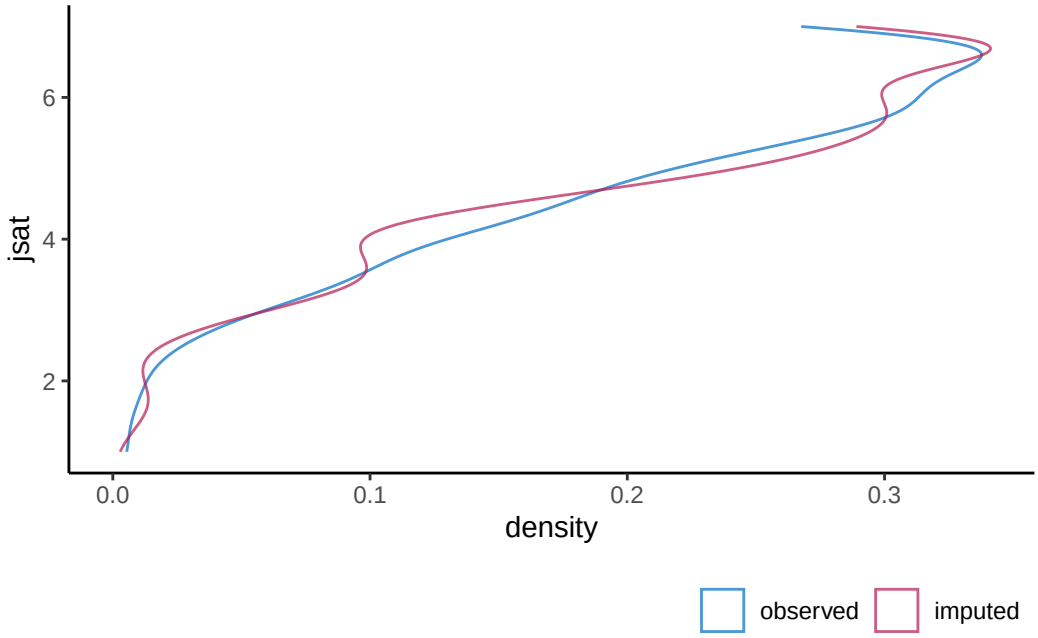
A boxplot comparing observed and imputed values for `jsat`.

```
missing_perceptions_jsat_data_multiimp %>%
  ggmice(aes(x = .imp, y = jsat)) +
  geom_jitter(height = 0, width = 0.25) +
  geom_boxplot(width = 0.5, size = 1, alpha = 0.75, outlier.shape = NA) +
  labs(x = "Imputation number")
```



A density plot comparing observed and imputed values for `jsat`.

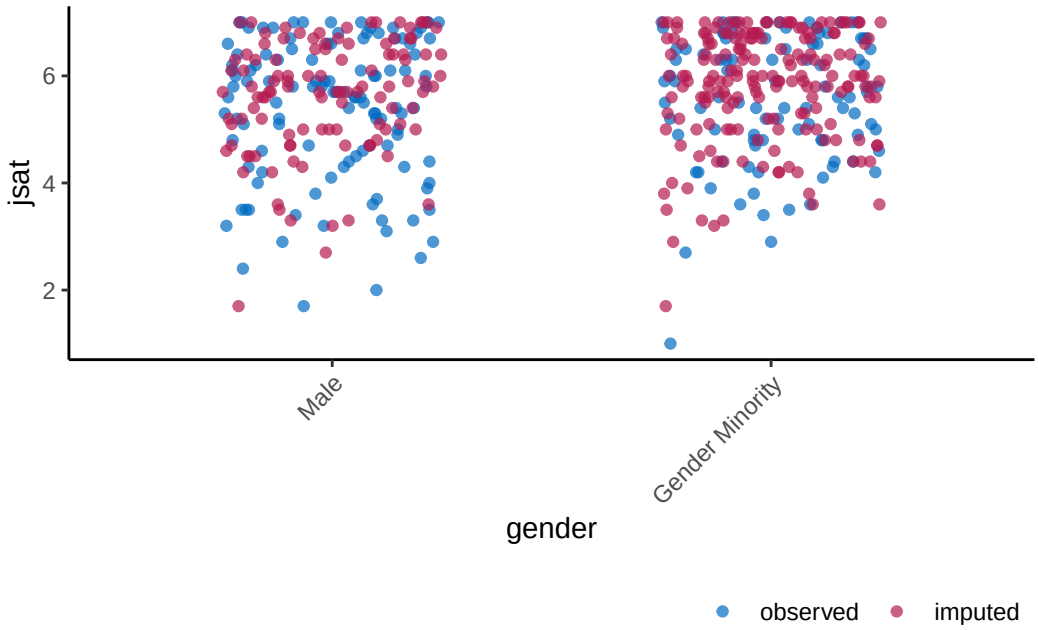
```
missing_perceptions_jsat_data_multiimp %>%
  ggmice(aes(y = jsat)) +
  geom_density()
```



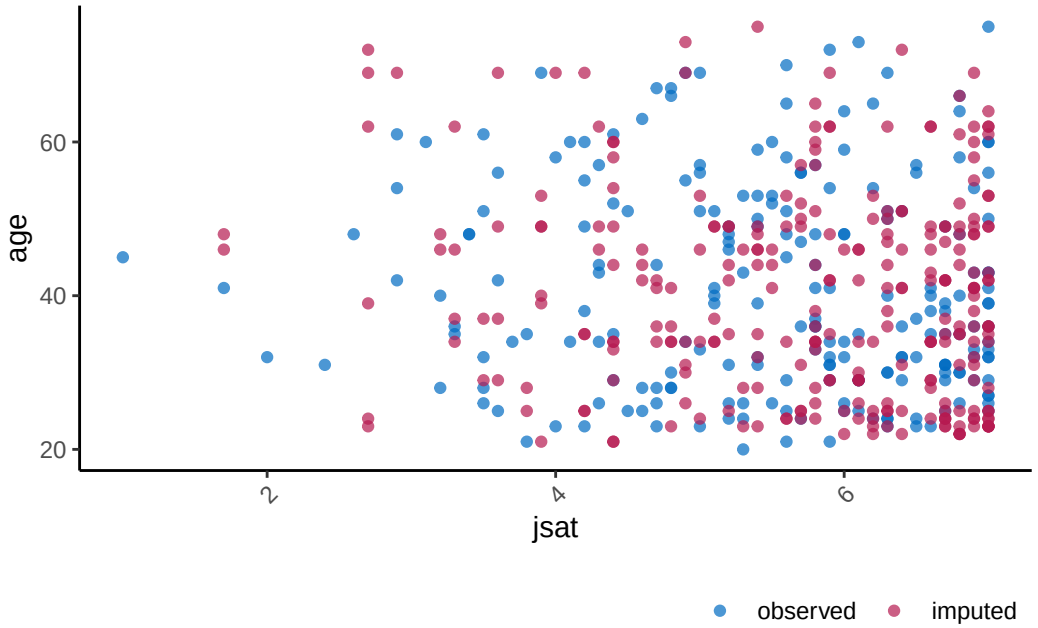
In the case of MAR, where missing values are dependent on other variables, we would want to look at observed vs imputed values for a variable of interest in relation to another variable. We can examine the observed vs imputed data points for jsat in relation with gender, age, race, and education

```
# jsat and gender
missing_perceptions_jsat_data_multiimp %>%
  ggmice(aes(x = gender, y = jsat)) +
  geom_jitter(height = 0, width = 0.25, seed = 1234) +
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust=1))
```

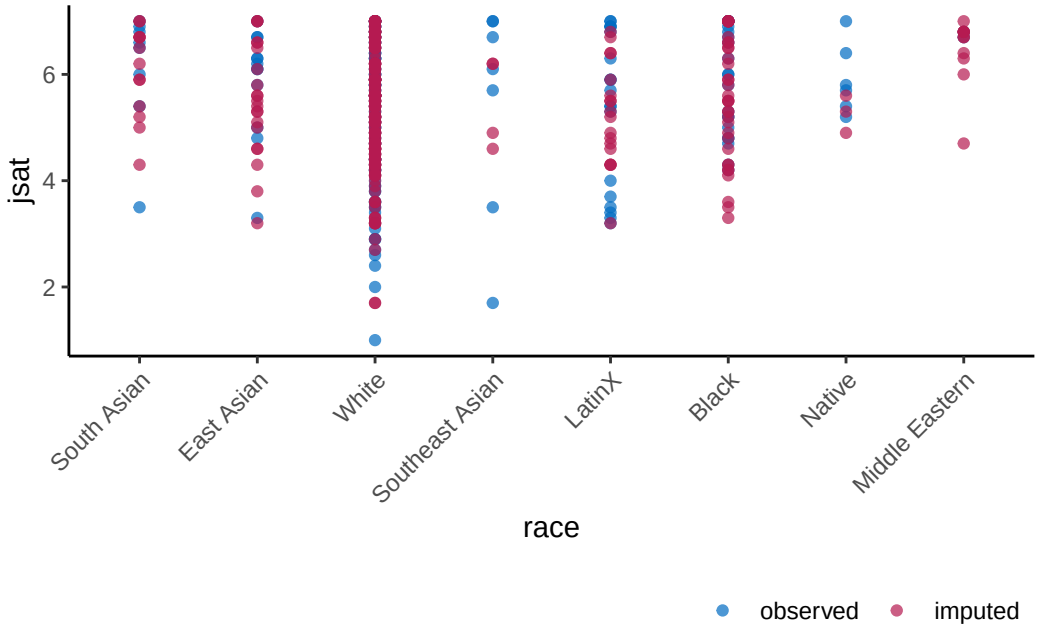
```
Warning in geom_jitter(height = 0, width = 0.25, seed = 1234): Ignoring unknown
parameters: `seed`
```



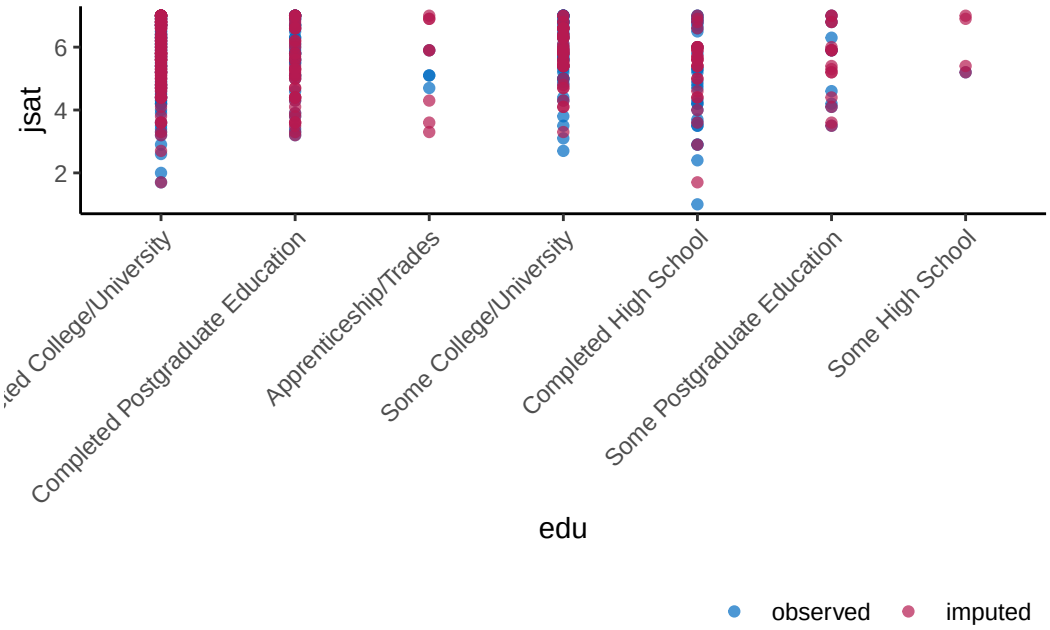

```
# jsat and age
missing_perceptions_jsat_data_multiimp %>%
  ggmice(aes(x = jsat, y = age)) +
  geom_point() +
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust=1))
```



```r
# jsat and race
missing_perceptions_jsat_data_multiimp %>%
  ggmice(aes(x = race, y = jsat)) +
  geom_point() +
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust=1))
```




```r
# jsat and education
missing_perceptions_jsat_data_multiimp %>%
  ggmice(aes(x = edu, y = jsat)) +
  geom_point() +
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust=1))
```



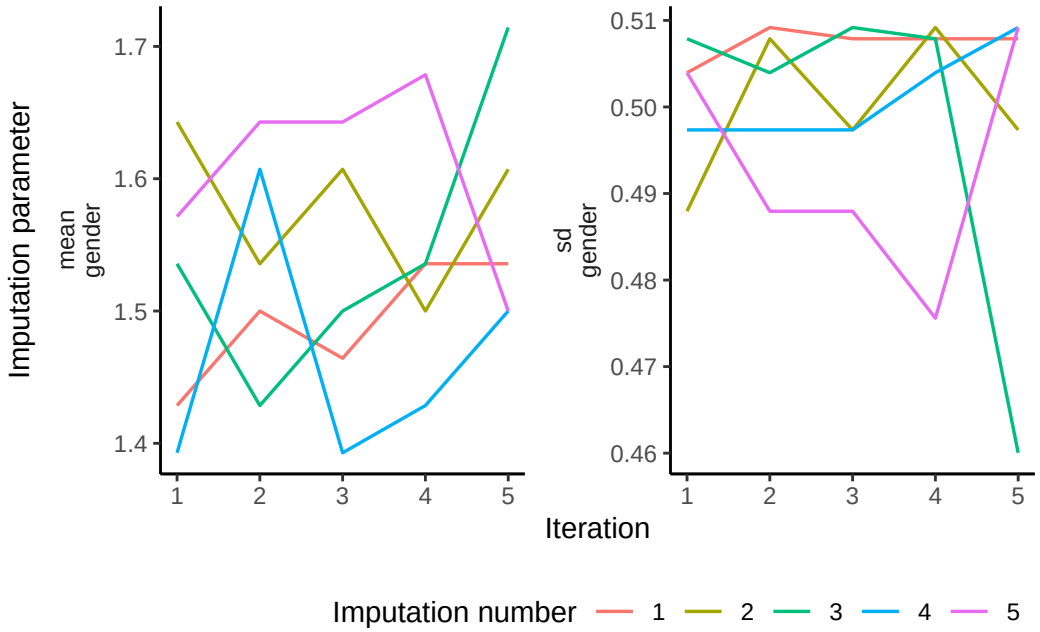
7.2 Convergence

> There is no clear-cut method for determining whether the MICE algorithm has converged. What is often done is to plot one or more parameters against the iteration number…On convergence, the different streams should be freely intermingled with each other, without showing any definite trends. Convergence is diagnosed when the variance between different sequences is no larger than the variance with each individual sequence (Buren & Groothuis-Oudshoorn, 2011, p. 37)

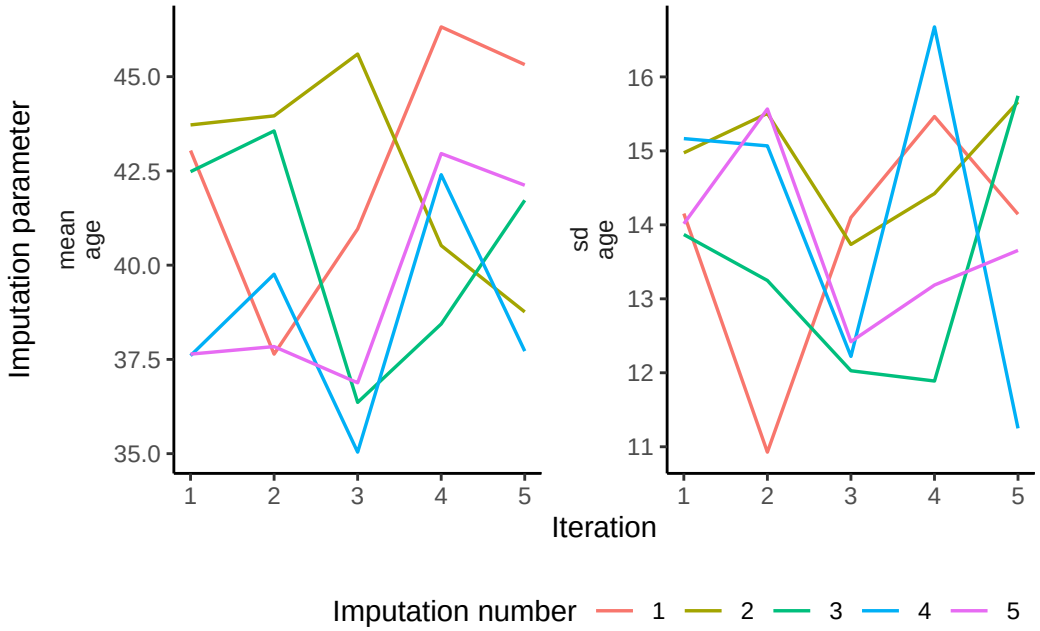
Trace plots are used to visually inspect the convergence of imputed values across iterations. They plot the values of the imputed data across each iteration for one or more imputed variables.

> **!** Important
>
> Each plot should ideally show the imputation values stabilizing as iterations increase. Fluctuating or divergent patterns suggest non-convergence.

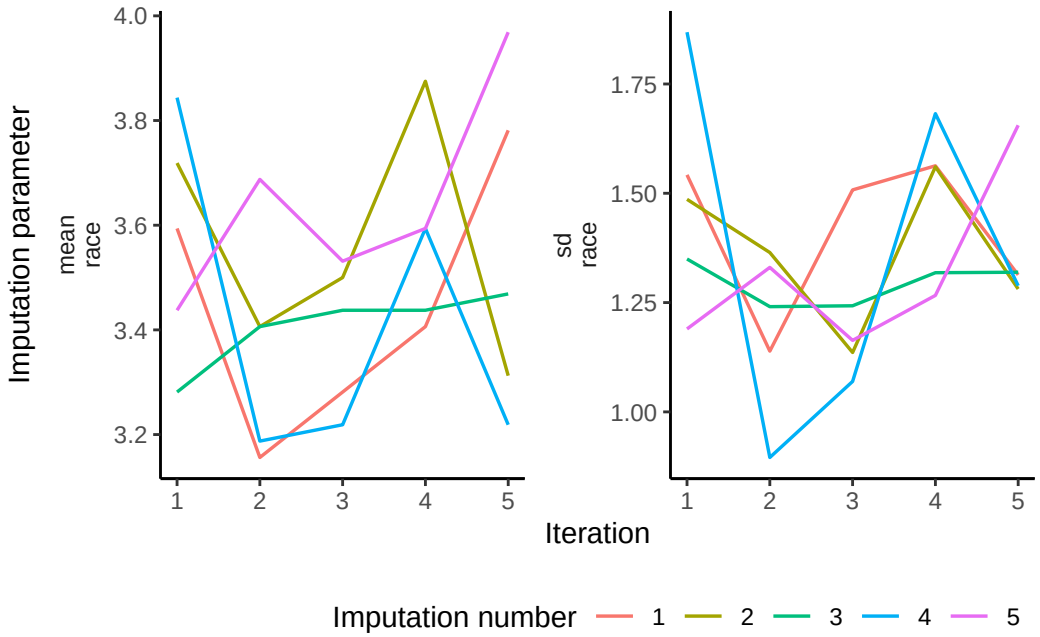
```
plot_trace(missing_perceptions_jsat_data_multiimp, "gender")
```



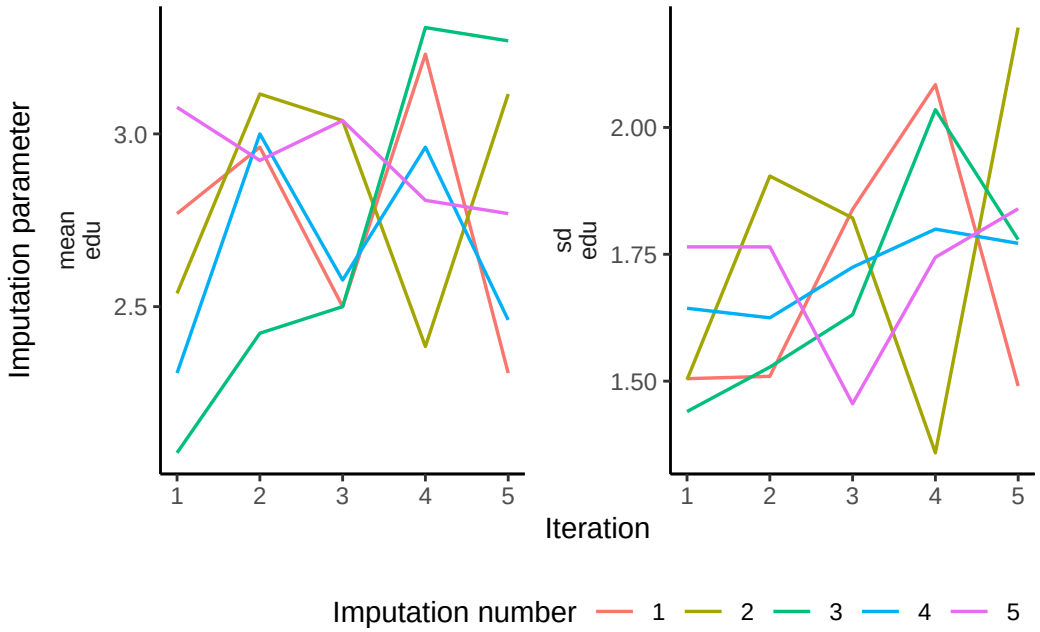

```
plot_trace(missing_perceptions_jsat_data_multiimp, "age")
```



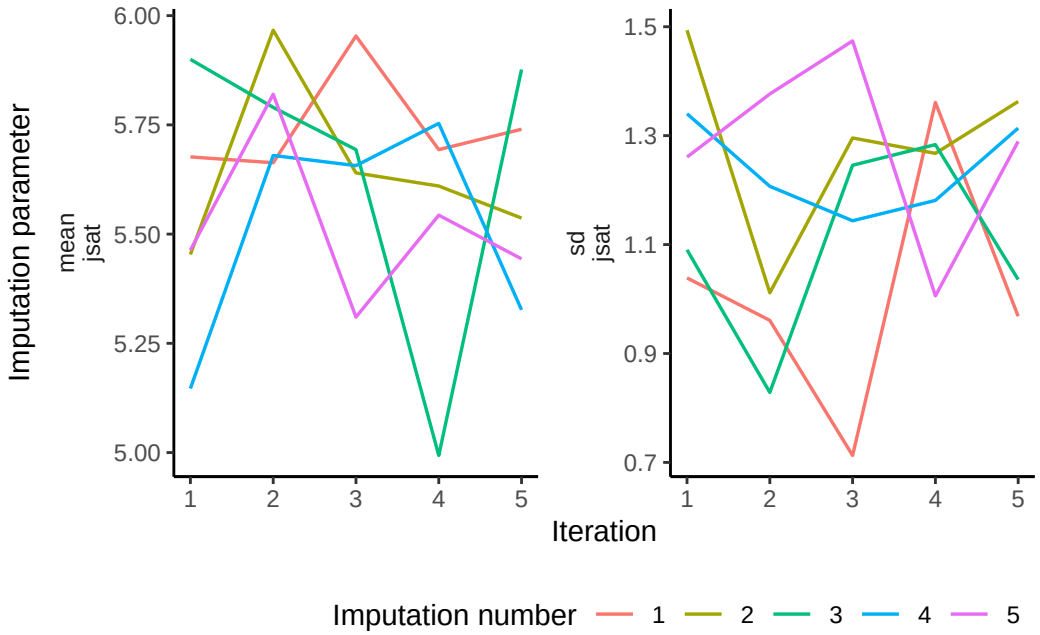

```
plot_trace(missing_perceptions_jsat_data_multiimp, "race")
```




```
plot_trace(missing_perceptions_jsat_data_multiimp, "edu")
```




```
plot_trace(missing_perceptions_jsat_data_multiimp, "jsat")
```


Imputation parameter
mean
jsat
6.00
5.75
5.50
5.25
5.00
sd
jsat
1.5
1.3
1.1
0.9
0.7
1
2
3
4
5
Iteration
Imputation number
1
2
3
4
5

8 Other Imputation Methods

We were only able to cover one imputation method during this workshop, however, there are other imputation methods. In this section, we will provide a brief overview of the other methods. Also, additional resources are provided below for more details on this topic.

- **Donor-based imputation**: For each observation with some incomplete data, we can replace them with data from other, complete observations. In other words, the complete observations that donate their data to the incomplete ones are the donors. There are three main donor-based imputation methods.
  - **Mean imputation**: One of the simplest imputation methods is the mean imputation. It boils down to simply replacing the missing values for each variable with the mean of its observed values.
  - **Hot-deck imputation**: This method simply replaces every missing value with the last observed value in the same variable
  - **k-Nearest-Neighbours imputation (kNN)**: Look for a chosen number of k other observations, or neighbours, that are similar to that observation. Then, replace the missing values with the aggregated values from the k donors. In order to choose most similar donors, we need a measure of similarity or distance, between observations.
- **Model-based imputation:** This model is used to predict the missing values. The general idea is to loop over the variables, for each of them create a model that explains it using the remaining variables. We then iterate through the variables multiple times, imputing the locations where the data were originally missing.
  - **Regression imputation:** Performs regression imputation according to the specified formula.
  - **Tree imputation:** A machine learning model to predict missing values. The algorithm uses decision trees to make new predictions.
- **Multiple imputation by bootstrapping**: Our workshop did not cover the other multiple imputation method, bootstrapping. First, take many bootstrap samples from the original data. Then, we impute each sample with method of choice. Next, perform some analysis or modelling on each of the many imputed data sets. Finally, when we obtain a single result from each bootstrap sample, we put these so-called bootstrap replicates together to form a distribution of results. We can then use the mean of this distribution as a point estimate or look at the quantiles to obtain confidence intervals.

9 Resources

9.1 DataCamp Courses

- Dealing with missing data in R
- Handling missing data with imputations in R

9.2 Textbooks

Kabacoff, R. I. (2022). *R in Action Data Analysis and graphics with R and Tidyverse*. Manning Publications. https://www.manning.com/books/r-in-action-third-edition

McKnight, P. E. (2007). *Missing data: A gentle introduction*. Guilford Press. https://www.amazon.ca/Missing-Data-Introduction-McKnight-2007-03-28/dp/B01JPRMOKM/ref=mp_s_a_1_2?crid=2ZNZ4DVTHTE86&dib=eyJ2IjoiMSJ9.aXvSeoPK9aeJAOk4f0bi2BC7kYDowG40Gx4NJXF2fHmc4nPBXu_J69etQczim0s7QLDkmPp0UBJgUKPW3EERn_VMbGtgHhGywfXz6h_PKJUsy5OMBciqQ-2IAIchLCtm.8I1DD_nnXQDK22IH7nLCMnuM9ywjfiNQNFx95OCjS4Q&dib_tag=se&keywords=missing+data+gentle+introduction&qid=1712591330&sprefix=missing+data+gentle+introduction,aps,66&sr=8-2

Little, R. J. A., & Rubin, D. B. (2019). *Statistical Analysis with Missing Data*. Wiley. https://www.amazon.ca/Statistical-Analysis-Missing-Roderick-Little/dp/0470526793/ref=mp_s_a_1_2?crid=34GEPLB6MMD8B&dib=eyJ2IjoiMSJ9.Gk834hLbMTvuHZHTGz64nLZ8YjofumZ4ayhP02–rB0lVbbWKDYXiyTMFHpTWwgnSoCR1IvY2JuLQ08v9kgkNA.1UbODQxyQzUmf9lyXdtx3Voy_RH4R7ev1cuGoB_CDH8&dib_tag=se&keywords=missing+data+knight&qid=1712591267&sprefix=missing+data+knight%2Caps%2C62&sr=8-2