Missing Data: Tutorial 1

Albert Varela & José Pina-Sánchez

Table of contents

Introduction

This first tutorial will introduce you to missing data from a simulation perspective. The objective here is to help you understand missing data mechanisms and their implications for data analysis and inference, without the complications inherent to real world datasets which often contain missing responses.

In addition to literature cited earlier in the lecture slides, such as Buuren (2018) or Enders (2022) , this tutorial draws on and follows closely the more detailed discussion of missing data simulation in Zhang (2023) (reproducible code available here), and the the articles demonstrating the function *ampute* in mice (Schouten, Lugtig, and Vink 2018).

We will use the following packages throughout:

```
library(MASS)
library(tidyverse)
library(mice)
library(ggmice)
library(knitr)
```

```
library(psych)
library(patchwork)
```

Simulating a random sample from a population

First of all, we begin our discussion of missing data by generating a synthetic dataset that represents a sample from a population. To do this we need to specify the data generation mechanism (number of variables, means and covariance matrix) as well as the sample size as detailed below:

- The sample dataset contains 10,000 cases measured across six different variables;
- These six variables have means equal to zero and are normally distributed;
- These six variables are moderately and positively correlated among themselves (r = 0.5)

```
set.seed(123)                                                          ①
mu <- rep(0,6)                                                         ②
Sigma <- matrix(c(1, 0.5, 0.5, 0.5, 0.5, 0.5,
                  0.5, 1, 0.5, 0.5, 0.5, 0.5,
                  0.5, 0.5, 1, 0.5, 0.5, 0.5,
                  0.5, 0.5, 0.5, 1, 0.5, 0.5,
                  0.5, 0.5, 0.5, 0.5, 1, 0.5,
                  0.5, 0.5, 0.5, 0.5, 0.5, 1),
                nrow=6, ncol=6)                                        ③
sample.size <- 10000                                                   ④
complete.data <- as_tibble(mvrnorm(sample.size, mu, Sigma))            ⑤
```

① Set seed for random number generator
② Define variable means equal to 0
③ Specify sample covariance matrix between variables (corr = 0.5)
④ Total number of observations (n = 10,000)
⑤ Define the simulated dataset based on sample size, variable means, and covariance matrix.

This is the resulting dataset:

```
complete.data |>
  print(n=10)
```

```
# A tibble: 10,000 x 6
       V1      V2      V3      V4      V5      V6
    <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
```

```
 1  0.602  0.487  -1.23   -0.762  -0.415  -1.25
 2 -0.650  0.282  -0.469   0.270  -0.133  -0.354
 3  1.95   1.09    1.12    0.577   2.25    0.151
 4  0.128 -0.582  -0.143  -0.297   1.44   -0.226
 5  0.268  0.0930 -0.343   1.02    0.324  -0.774
 6  0.885  2.94    1.12    0.900   1.84    0.176
 7  2.04  -0.527  -0.441   0.801   0.145   0.0953
 8 -0.883 -1.30   -1.50    0.0356 -1.86   -0.297
 9 -1.27  -1.08   -0.0317 -0.175  -0.0941 -0.499
10 -0.854  0.0430  0.515  -0.842  -0.845  -0.0591
# i 9,990 more rows
```

These are the basic descriptive statistics for our sample:

```
complete.data |>
  describe()
```

```
   vars     n  mean   sd median trimmed  mad   min  max range  skew kurtosis
V1    1 10000 -0.01 1.00  -0.02   -0.01 1.02 -3.42 4.20  7.62  0.01    -0.05
V2    2 10000  0.00 1.00   0.00    0.00 0.99 -4.17 4.12  8.29  0.00     0.01
V3    3 10000 -0.01 0.99   0.00   -0.01 1.00 -3.36 3.78  7.13 -0.01    -0.06
V4    4 10000  0.00 1.00   0.01    0.00 0.99 -3.73 3.52  7.25 -0.01     0.04
V5    5 10000  0.01 0.99   0.00    0.01 1.00 -3.86 3.68  7.54 -0.03    -0.04
V6    6 10000  0.00 1.01  -0.01   -0.01 1.01 -3.96 3.75  7.72  0.03     0.00
     se
V1 0.01
V2 0.01
V3 0.01
V4 0.01
V5 0.01
V6 0.01
```

Simulating missingness

We begin our practical examination of missing data by applying a Missing Completely at Random mechanism to our simulated data. Recall that MCAR refers to cases where missingness is unrelated to either the observed or the unobserved part of the data - in other words, that the probability of missing information or having an empty cell in the dataset is strictly random.

$$Pr(Y\text{miss}|Y, X) = Pr(Y\text{miss})$$

If we are collecting data on incomes, the probability of non-response does not vary either by, say, education, occupation, age or geographical area, or by the income variable itself. All units have basically the same probability of being missing in any given variable.

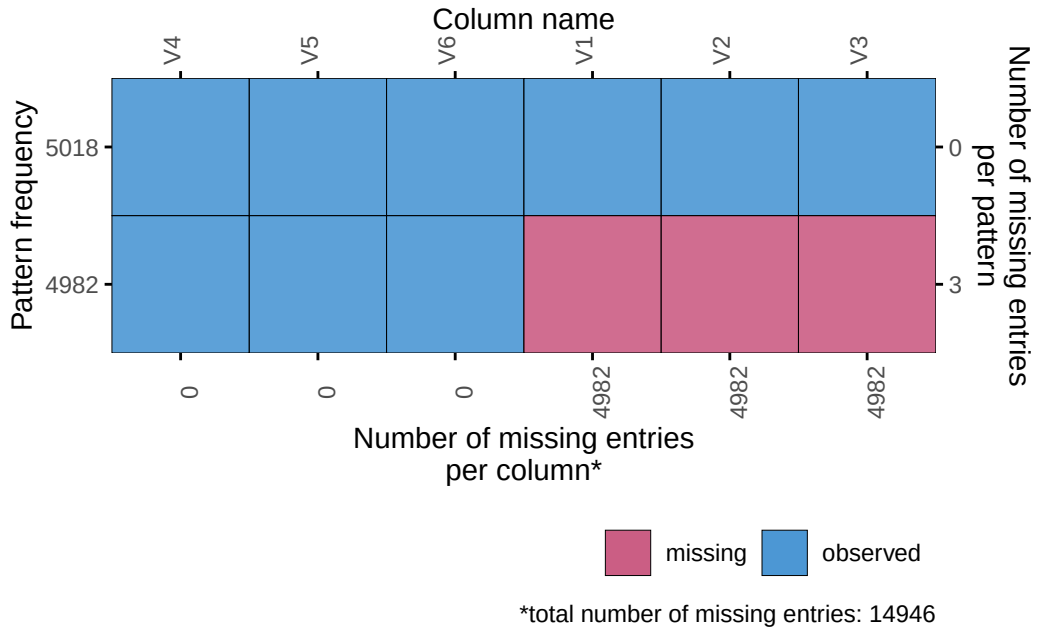
To simulate this mechanism we need to decide the share of missing values we'd like to introduce, for which or how many variables, and then generate a simulated missing data pattern based on a MCAR mechanism that can then be applied to the our simulated dataset.

```
missing.percentage <- 0.5                                                    ①
var.with.missing <- 1:3                                                      ②
mcar.min.pattern <- complete.data                                            ③
ind <- as.logical(rbinom(sample.size, 1, missing.percentage))                ④
mcar.min.pattern[ind, var.with.missing] <- NA                                ⑤
```

① Specify the share of missing data we want (50%)
② Specify the variables that will have missing data (V1, V2 and V3)
③ Generate the new dataset in which missing values will be imputed
④ Simulate MCAR pattern by sampling 10,000 0/1 values from a binomial distribution - such that 50% are missing - and then transform them into a FALSE/TRUE logical vector.
⑤ Apply MCAR mechanism by assigning NA to the first three variables in our data where the simulate MCAR pattern returns a 0 value.

The resulting missing pattern, plotted using the *plot_pattern()* function from *ggmice* is as follows:

```
plot_pattern(
  mcar.min.pattern,
  square = TRUE,
  rotate = TRUE
)
```



Notice that because we specified a single pattern of missing data to be applied to the three variables, the result is that those cases with missing values on V1 also have missing values on V2 and V3, but valid values on V4, V5 and V6.

These are the basic descriptive statistics for our data after the missingness has been applied:

```
mcar.min.pattern |>
  describe()
```

```
   vars     n mean   sd median trimmed  mad   min  max range  skew kurtosis
V1    1  5018 0.01 1.01  -0.01    0.00 1.02 -3.39 3.52  6.91  0.05    -0.09
V2    2  5018 0.01 1.00   0.02    0.02 0.99 -4.17 4.12  8.29 -0.02     0.08
V3    3  5018 0.01 0.99   0.02    0.01 1.01 -3.36 3.44  6.79 -0.02    -0.14
V4    4 10000 0.00 1.00   0.01    0.00 0.99 -3.73 3.52  7.25 -0.01     0.04
V5    5 10000 0.01 0.99   0.00    0.01 1.00 -3.86 3.68  7.54 -0.03    -0.04
V6    6 10000 0.00 1.01  -0.01   -0.01 1.01 -3.96 3.75  7.72  0.03     0.00
     se
V1 0.01
V2 0.01
V3 0.01
V4 0.01
V5 0.01
V6 0.01
```

Key things to notice here:

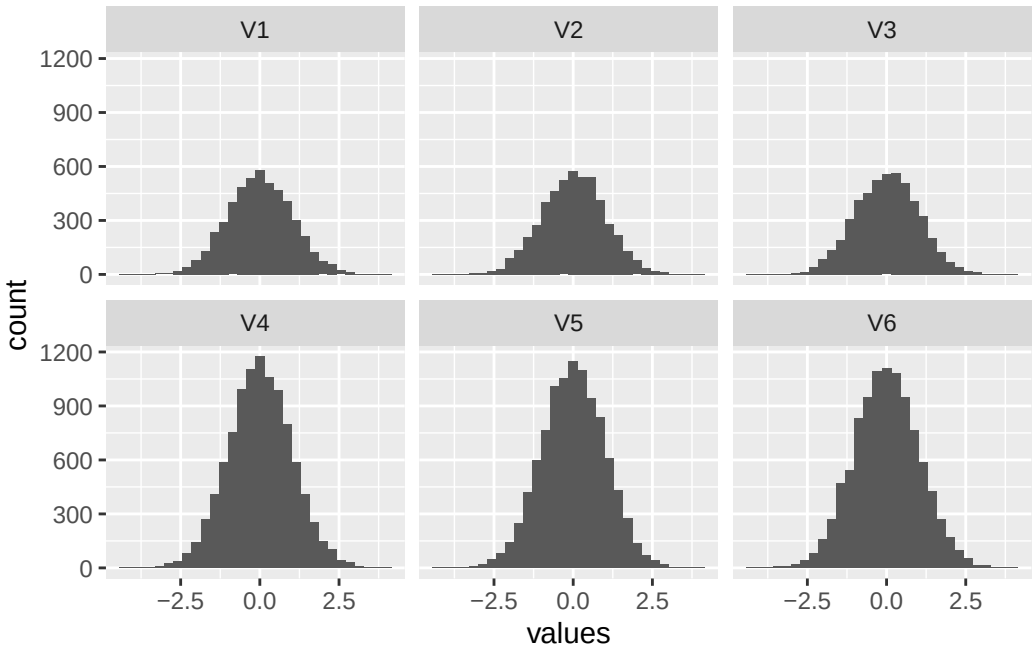
1. MCAR has NOT biased estimates of the mean and standard deviations of the variables that were subject to the imputation of missing values
2. MCAR has led to larger standard errors due to reduction in effective sample size

And we can also explore the distribution of the variables

```
mcar.min.pattern |>
  pivot_longer(starts_with("V"),
               names_to = "variable",
               values_to = "values") |>
  ggplot(aes( x = values)) +
  geom_histogram() +
  facet_wrap(~variable)
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
Warning: Removed 14946 rows containing non-finite values (`stat_bin()`).
```



For sake of completion, let's now simulate different MCAR missingness patterns for each one of the variables, that is, the probability of missingness in any one of the three variables is

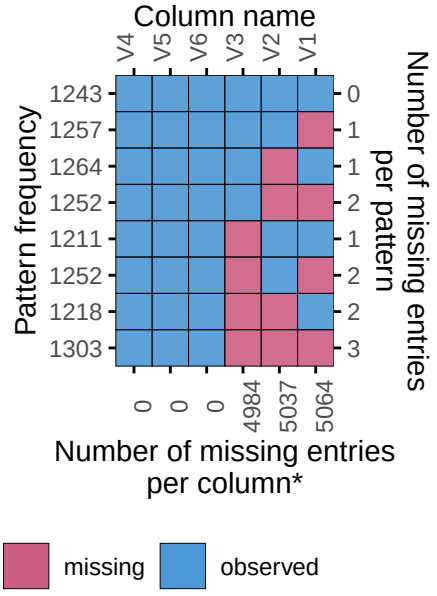
completely unrelated to the missingness in any the other two. In order to achieve this we will use the same function to draw a random sample from a binomial distribution and then assign missing values (NA) that will be repeated three times, one per each of the variables we are targeting for missing value imputations:

```
mcar.max.pattern <- complete.data
for(i in var.with.missing){                                                    ①
  ind <- as.logical(rbinom(sample.size, 1, missing.percentage))
  mcar.max.pattern[ind,i] <- NA
}
```

① This *for()* loop function repeats the simulation of the missing pattern for each one of the variables being imputed with missing values.

The *plot.pattern()* function will show graphically the main missingness patterns in the resulting dataset. Here, unlike the previous MCAR simulation, we can see how each variable has its own missingness pattern, resulting in cases that have missing data for just one variable or for a combination of two or three different variables.

```
plot_pattern(
  mcar.max.pattern,
  square = TRUE,
  rotate = TRUE
)
```



*total number of missing entries: 15085

Notice, however, that having different missingness patterns across different variables does not in any way alter the fact that the empty cells remain MCAR and therefore we should expect unbiased estimates of the main quantities of interest (mean, standard deviations, and so on) exactly as before.

```
mcar.max.pattern |>
  describe()
```

```
   vars     n  mean   sd median trimmed  mad   min  max range  skew kurtosis
V1    1  4936 -0.01 1.00  -0.02   -0.01 1.01 -3.39 4.20  7.59  0.06    -0.05
V2    2  4963  0.00 1.01   0.01    0.00 1.00 -3.25 4.12  7.37 -0.04    -0.05
V3    3  5016  0.00 0.99   0.00    0.00 1.00 -3.35 3.44  6.79 -0.01    -0.11
V4    4 10000  0.00 1.00   0.01    0.00 0.99 -3.73 3.52  7.25 -0.01     0.04
V5    5 10000  0.01 0.99   0.00    0.01 1.00 -3.86 3.68  7.54 -0.03    -0.04
V6    6 10000  0.00 1.01  -0.01   -0.01 1.01 -3.96 3.75  7.72  0.03     0.00
     se
V1 0.01
V2 0.01
V3 0.01
V4 0.01
V5 0.01
V6 0.01
```

Using *ampute()* for simulating missingness

Simulating simple scenarios of missing data patterns can be done in R with relatively uncomplicated code, as seen in the previous section. Missingness, however, can take many different forms depending on a variety of parameters that can be quite cumbersome to code, particularly when we begin to consider incomplete data subject to MAR and MNAR mechanisms.

In the rest of this tutorial, we will use the function *ampute()* from the package *mice* to simulate missing data in a more systematic and clear way. This function and options is discussed at length in Schouten, Lugtig, and Vink (2018) and features prominently in a very interesting simulation paper by Schouten and Vink (2021). What follows adapts code from this vignette.

First, we will create another simulated dataset, with code that should look familiar from the earlier simulation:

```
library(mvtnorm)
set.seed(123)
sigma <- matrix(data = c(1, 0.5, 0.5, 1),                    ①
                ncol = 2)
```

```
simdata <- rmvnorm(n = 10000,                                    ②
                   mean = c(8, 3),                               ③
                   sigma = sigma) %>%
                       as_tibble() %>%
                       rename(x = V1, z = V2) %>%
                       mutate(y = 6 * x + 3 * z + rnorm(1000))   ④
describe(simdata)
```

① 2 variables x and z with a correlation of 0.5
② 10,000 cases randomly drawn from a multivariate normal
③ Means equal to 8 and 3 with correlations as specified before
④ Define new variable y as a function of x and z plus some randomly distributed noise

```
  vars     n  mean sd median trimmed  mad   min   max range  skew kurtosis   se
x    1 10000  8.00  1   7.99    8.00 1.00  3.77 12.00  8.23 -0.01     0.03 0.01
z    2 10000  2.99  1   2.99    2.99 1.00 -0.80  6.77  7.56  0.03     0.02 0.01
y    3 10000 56.93  8  56.91   56.94 7.91 22.02 88.92 66.90 -0.01     0.06 0.08
```

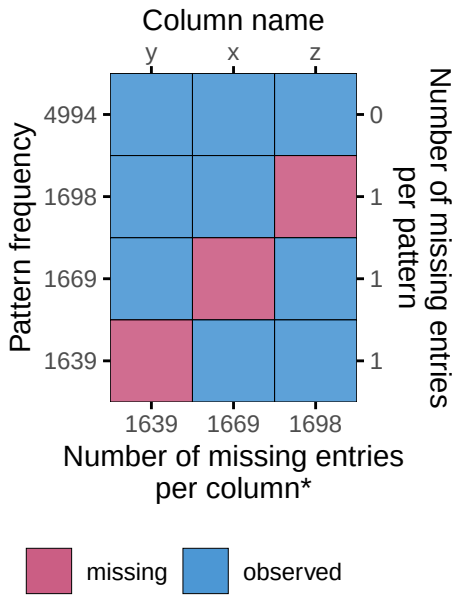
Simulating MCAR

To simulate missing completely at random values for a simulated dataset we use the function *ampute()* using the option *mech = "MCAR"* to indicate a missing completely at random mechanism:

```
mcar_sim <-
  simdata |>
  ampute(mech = "MCAR")
```

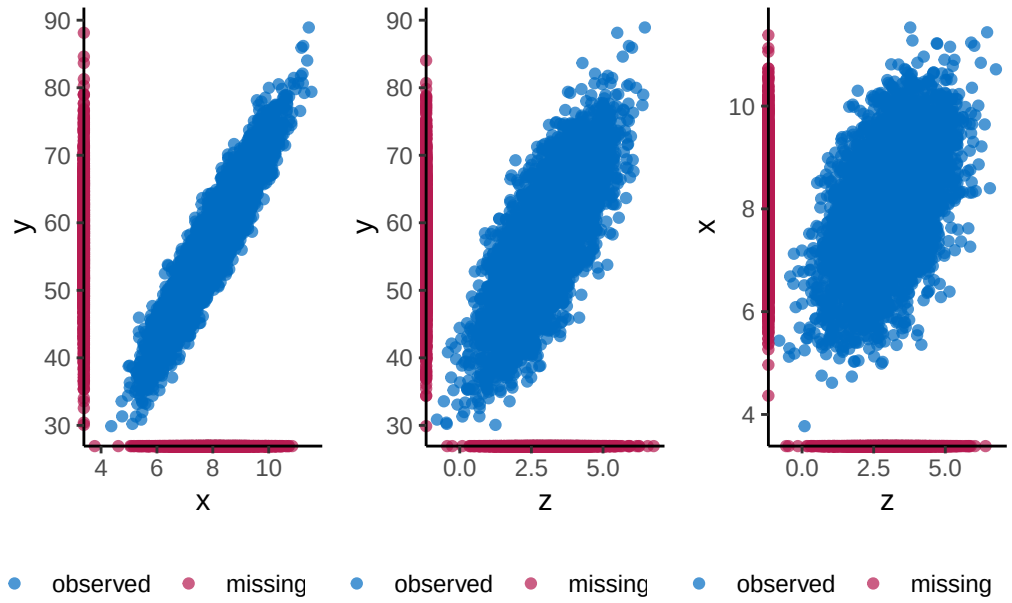
Notice that the resulting object mcar_sim is not a dataframe as such, but it does contain the dataset within and be called as *mcar_sim$amp*. Let's examine the resulting missingness pattern.

```
plot_pattern(mcar_sim$amp)
```



Plotting the missing values along the axis of a scatterplot shows missingness is spread evenly across the values of x and y.

```
plot_a <-ggmice(mcar_sim$amp, aes(x, y)) +
  geom_point()
plot_b <-ggmice(mcar_sim$amp, aes(z, y)) +
  geom_point()
plot_c <-ggmice(mcar_sim$amp, aes(z, x)) +
  geom_point()
plot_a + plot_b + plot_c
```



We can compare the original dataset to the incomplete one we just created:

```
describe(simdata)
```

```
  vars     n  mean sd median trimmed  mad   min   max range  skew kurtosis   se
x    1 10000  8.00  1   7.99    8.00 1.00  3.77 12.00  8.23 -0.01     0.03 0.01
z    2 10000  2.99  1   2.99    2.99 1.00 -0.80  6.77  7.56  0.03     0.02 0.01
y    3 10000 56.93  8  56.91   56.94 7.91 22.02 88.92 66.90 -0.01     0.06 0.08
```

```
describe(mcar_sim$amp)
```

```
  vars    n  mean sd median trimmed  mad   min   max range skew kurtosis   se
x    1 8331  7.99  1   7.98    7.99 0.99  3.77 11.53  7.76 0.00    -0.01 0.01
z    2 8302  2.99  1   2.99    2.99 1.00 -0.80  6.77  7.56 0.02     0.04 0.01
y    3 8361 56.90  8  56.86   56.91 7.90 29.88 88.92 59.04 0.01     0.04 0.09
```

Again, no evidence of bias, as we should expect from MCAR.

So far we have used the default options *ampute()*. Now, still under MCAR, we can use a number of option to simulate more complex scenarios such as:

- prop: proportion of incomplete rows (default: 50%)

- patterns: different combinations of missingness across different variables (default: 1 pattern per variable)
- freq: the relative size of each pattern (default: same frequency per pattern)

For instance:

- impute missing values in 70% of the rows,
- Allow missingness to occur in simultaneously in x and z,
- Force missingness pattern for a single variable tobe three times more frequent than for two variables.

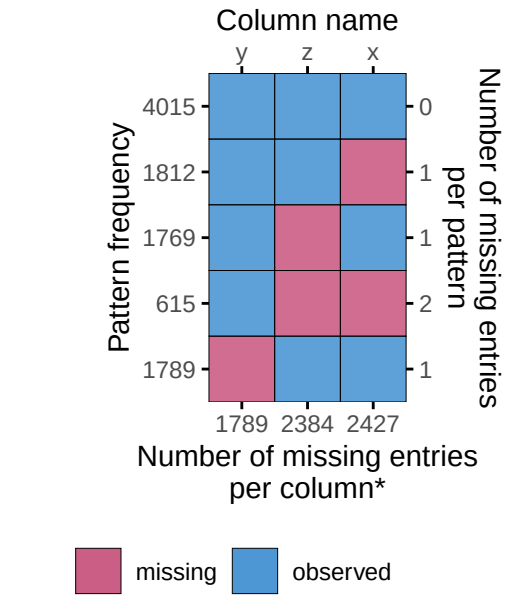
```
my_patterns <- mcar_sim$patterns
my_patterns <- rbind(my_patterns, c(0, 0, 1))                    ①
my_freq <- c(.3, .3, .3, .1)                                     ②

mcar_sim_cust <-
  simdata |>
  ampute( prop = 0.6,
          patterns = my_patterns,
          freq = my_freq,
          mech = "MCAR")
```

① Add a fourth missingness pattern #4 such that x and z (but not y) can be missing at the same time

② Specify that the frequencies for patterns 1 to 3 for missingness in a single variable is three times greater than for the fourth pattern

Notice that the resulting object mcar_sim is not a dataframe as such, but it does contain the dataset within and be called as *mcar_sim$amp*. Let's examine the resulting missingness pattern.

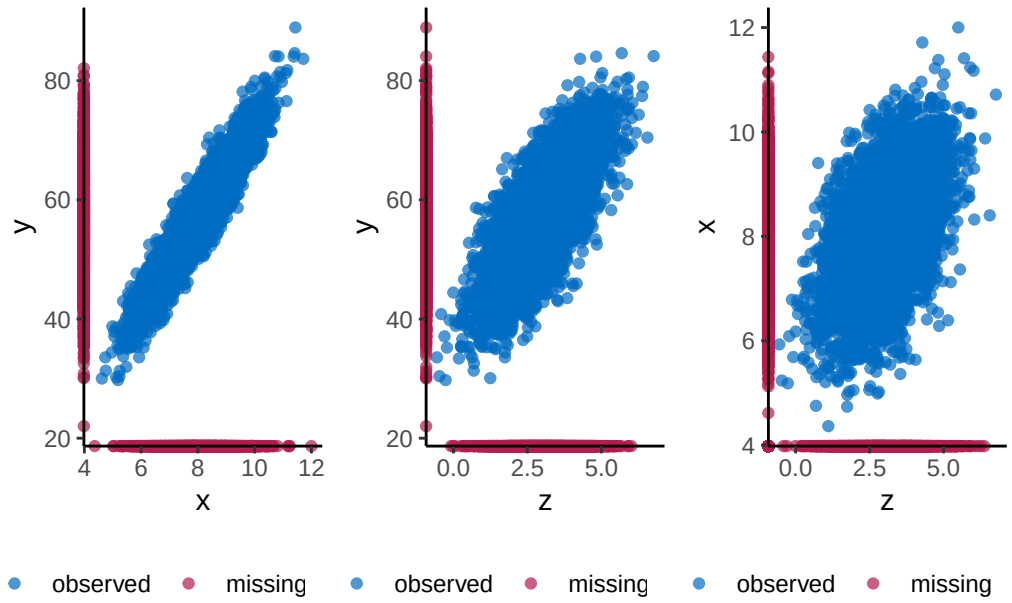
```
plot_pattern(mcar_sim_cust$amp)
```



*total number of missing entries: 6600

Plotting the missing values along the axis of a scatterplot shows missingness is spread evenly across the values of x and y.

```
plot_a <-ggmice(mcar_sim_cust$amp, aes(x, y)) +
  geom_point()
plot_b <-ggmice(mcar_sim_cust$amp, aes(z, y)) +
  geom_point()
plot_c <-ggmice(mcar_sim_cust$amp, aes(z, x)) +
  geom_point()
plot_a + plot_b + plot_c
```



We can compare the original dataset to the incomplete one we just created:

```
describe(simdata)
```

```
  vars     n  mean sd median trimmed  mad   min   max range  skew kurtosis   se
x    1 10000  8.00  1   7.99    8.00 1.00  3.77 12.00  8.23 -0.01     0.03 0.01
z    2 10000  2.99  1   2.99    2.99 1.00 -0.80  6.77  7.56  0.03     0.02 0.01
y    3 10000 56.93  8  56.91   56.94 7.91 22.02 88.92 66.90 -0.01     0.06 0.08
```

```
describe(mcar_sim$amp)
```

```
  vars    n  mean sd median trimmed  mad   min   max range skew kurtosis   se
x    1 8331  7.99  1   7.98    7.99 0.99  3.77 11.53  7.76 0.00    -0.01 0.01
z    2 8302  2.99  1   2.99    2.99 1.00 -0.80  6.77  7.56 0.02     0.04 0.01
y    3 8361 56.90  8  56.86   56.91 7.90 29.88 88.92 59.04 0.01     0.04 0.09
```

```
describe(mcar_sim_cust$amp)
```

```
  vars    n  mean   sd median trimmed  mad   min   max range  skew kurtosis
x    1 7573  8.00 1.00   8.01    8.00 0.99  4.36 12.00  7.64  0.00     0.01
z    2 7616  2.98 0.99   2.98    2.98 1.00 -0.56  6.77  7.32  0.03    -0.02
```

```
y    3 8211 56.89 8.01  56.90   56.90 7.89 22.02 88.92 66.90 -0.01     0.05
    se
x 0.01
z 0.01
y 0.09
```

Again, no evidence of bias, as we should expect from MCAR, even with the more complex specification.

Simulating MAR

Recall that MAR refers to situations where missingness is related to the observed part of the data - in other words, that the probability of units having an empty cell in a given variable is a function of the observed cells in other variables.

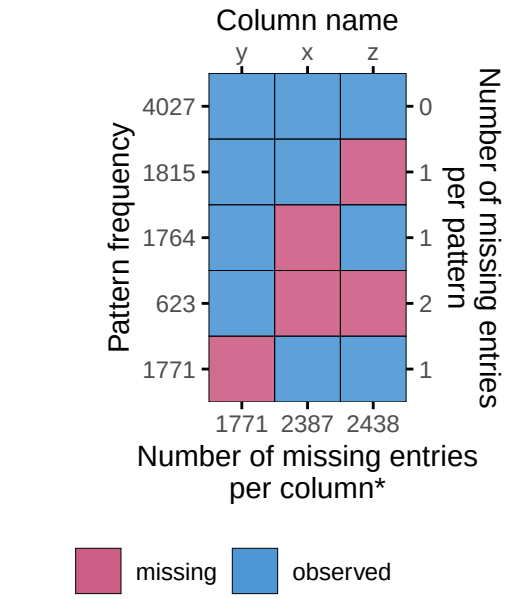
In addition the overall share, patterns, and frequency of patterns of missing data, we also need to consider the relative importance or *weight* of each variable in determine missingness on another variable as well as the *type* of missigness - i.e. how non-response varies as a function of the combined values of x.

Let's work with the defaults on those two options though first:

```
mar_sim <-
  simdata |>
  ampute( prop = 0.6,
          patterns = my_patterns,
          freq = my_freq,
          mech = "MAR")
```

Let's examine the resulting missingness pattern.

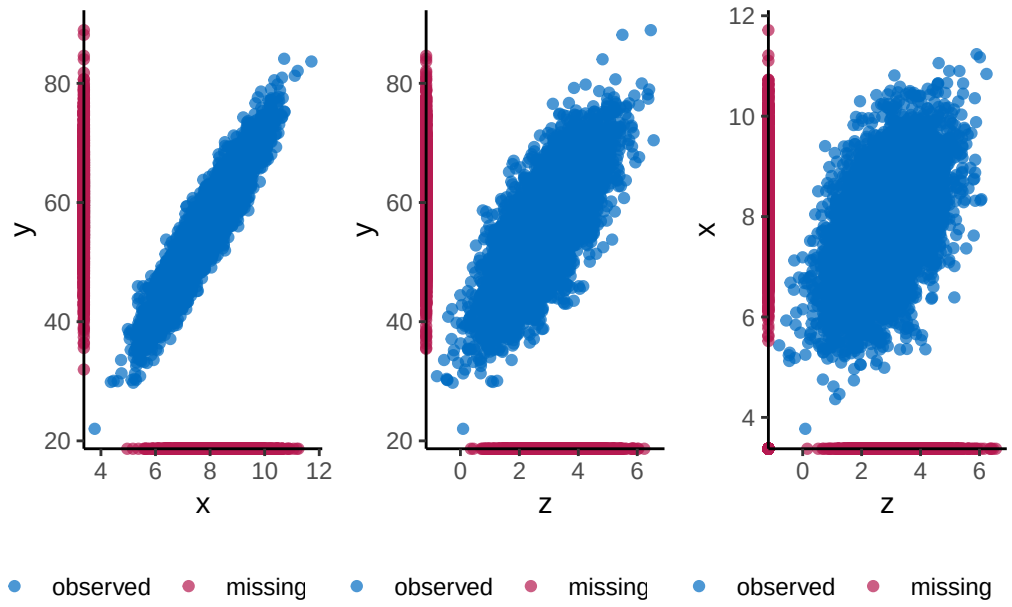
```
plot_pattern(mar_sim$amp)
```



*total number of missing entries: 6596

Plotting the missing values along the axis of a scatterplot shows missingness is more narrowly than before across the values of x and y.

```
plot_a <-ggmice(mar_sim$amp, aes(x, y)) +
  geom_point()
plot_b <-ggmice(mar_sim$amp, aes(z, y)) +
  geom_point()
plot_c <-ggmice(mar_sim$amp, aes(z, x)) +
  geom_point()
plot_a + plot_b + plot_c
```



We can compare the original dataset to the incomplete one we just created:

```
describe(simdata)
```

```
  vars     n  mean sd median trimmed  mad   min   max range  skew kurtosis   se
x    1 10000  8.00  1   7.99    8.00 1.00  3.77 12.00  8.23 -0.01     0.03 0.01
z    2 10000  2.99  1   2.99    2.99 1.00 -0.80  6.77  7.56  0.03     0.02 0.01
y    3 10000 56.93  8  56.91   56.94 7.91 22.02 88.92 66.90 -0.01     0.06 0.08
```

```
describe(mar_sim$amp)
```

```
  vars    n  mean   sd median trimmed  mad   min   max range skew kurtosis   se
x    1 7613  7.90 1.00   7.90    7.90 0.99  3.77 11.71  7.94 0.00     0.00 0.01
z    2 7562  2.91 1.00   2.90    2.91 0.99 -0.80  6.56  7.35 0.05     0.06 0.01
y    3 8229 56.43 8.05  56.33   56.42 7.96 22.02 88.92 66.90 0.02     0.05 0.09
```

Now we can begin to see some signs of bias, with means slightly lower than with the full data or the MCAR incomplete dataset.

We can introduce more complexity here still within a MAR framework:

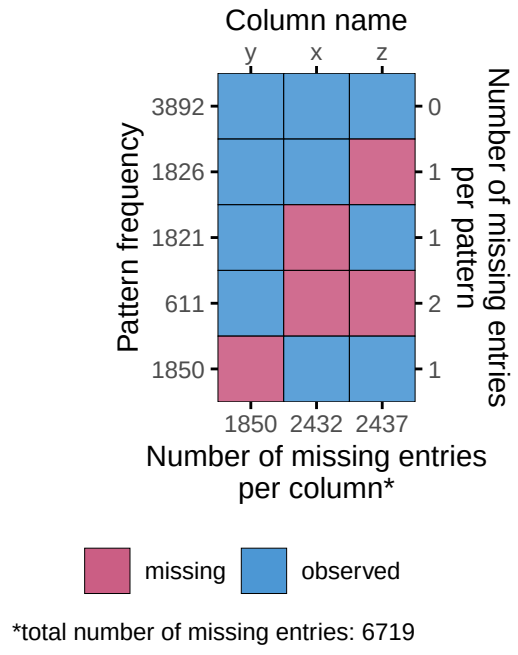
- Let's give z a greater weight in determining missingness in patterm 1 (default: equal weights)

- Let's force the probability of missingness to be higher for cases with a higher combined weight

```
my_weights <- mar_sim$weights
my_weights[1,] <- c(0,3,1)

mar_sim_right <-
  simdata |>
  ampute( prop = 0.6,
          patterns = my_patterns,
          freq = my_freq,
          mech = "MAR",
          weights = my_weights,
          type = "RIGHT")
```

Let's examine the resulting missingness pattern.

```
plot_pattern(mar_sim_right$amp)
```



*total number of missing entries: 6719

We can compare the original dataset to the incomplete one we just created:

```
describe(simdata)
```

```
  vars     n  mean sd median trimmed  mad   min   max range  skew kurtosis   se
x    1 10000  8.00  1   7.99    8.00 1.00  3.77 12.00  8.23 -0.01     0.03 0.01
z    2 10000  2.99  1   2.99    2.99 1.00 -0.80  6.77  7.56  0.03     0.02 0.01
y    3 10000 56.93  8  56.91   56.94 7.91 22.02 88.92 66.90 -0.01     0.06 0.08
```

```
describe(mar_sim$amp)
```

```
  vars    n  mean   sd median trimmed  mad   min   max range skew kurtosis   se
x    1 7613  7.90 1.00   7.90    7.90 0.99  3.77 11.71  7.94 0.00     0.00 0.01
z    2 7562  2.91 1.00   2.90    2.91 0.99 -0.80  6.56  7.35 0.05     0.06 0.01
y    3 8229 56.43 8.05  56.33   56.42 7.96 22.02 88.92 66.90 0.02     0.05 0.09
```

```
describe(mar_sim_right$amp)
```

```
  vars    n  mean   sd median trimmed  mad   min   max range  skew kurtosis
x    1 7568  7.93 1.00   7.91    7.93 1.00  3.77 12.00  8.23 -0.01    -0.01
z    2 7563  2.93 1.00   2.92    2.93 1.00 -0.80  6.56  7.35  0.03     0.02
y    3 8150 56.35 8.04  56.31   56.34 7.83 22.02 88.92 66.90  0.04     0.13
    se
x 0.01
z 0.01
y 0.09
```

Let's compare this with a mechanism that increases the probability of missingness for cases in the lower end of the combined weights (MAR_LEFT):

```
mar_sim_left <-
  simdata |>
  ampute( prop = 0.6,
          patterns = my_patterns,
          freq = my_freq,
          mech = "MAR",
          weights = my_weights,
          type = "LEFT")
describe(mar_sim_left$amp)
```

```
  vars    n  mean   sd median trimmed  mad   min   max range  skew kurtosis
x    1 7581  8.08 1.00   8.08    8.08 1.00  4.36 12.00  7.64  0.00     0.02
```

```
z    2 7595  3.07 1.01   3.07    3.07 1.02 -0.80  6.77  7.56  0.01     0.02
y    3 8265 57.48 8.01  57.52   57.51 7.96 22.02 88.92 66.90 -0.02     0.04
    se
x 0.01
z 0.01
y 0.09
```

The bias is now upwards as a result of dropping values for those cases with lower combined weights. Let's compare this with a mechanism that increases the probability of missingness in the middle combined weights:

```
mar_sim_mid <-
  simdata |>
  ampute( prop = 0.6,
          patterns = my_patterns,
          freq = my_freq,
          mech = "MAR",
          weights = my_weights,
          type = "MID")
describe(mar_sim_mid$amp)
```

```
  vars    n  mean   sd median trimmed  mad   min   max range  skew kurtosis
x    1 7619  7.99 1.03   7.98    7.99 1.03  4.36 12.00  7.64  0.02    -0.02
z    2 7553  3.00 1.03   3.00    3.00 1.04 -0.80  6.77  7.56  0.02    -0.03
y    3 8205 56.88 8.26  56.82   56.90 8.23 22.02 88.92 66.90 -0.01     0.00
    se
x 0.01
z 0.01
y 0.09
```

There little bias in the mean now (we have after all removed observations in the middle) but as a result the standard deviation has increased!

Simulating MNAR

We end this tutorial by simulating a MNAR mechanism. We will not specify weights this time as all we are interested is in producing missing values for a given variable such that the non-response is determined by the unobserved values in that very same variable:

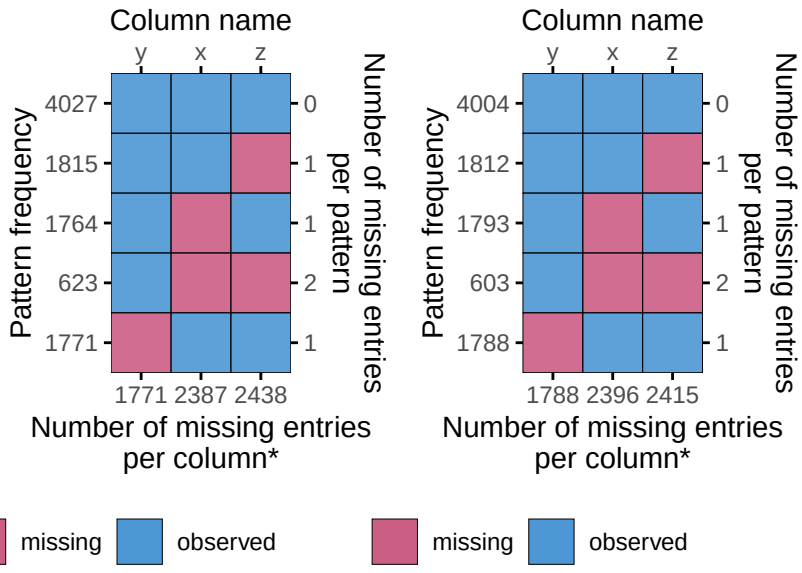
```
mnar_sim <-
  simdata |>
  ampute( prop = 0.6,
          patterns = my_patterns,
          freq = my_freq,
          mech = "MNAR")
describe(mnar_sim$amp)
```

```
  vars    n  mean   sd median trimmed  mad   min   max range skew kurtosis   se
x    1 7604  7.90 1.00   7.89    7.90 0.99  3.77 11.71  7.94 0.03     0.05 0.01
z    2 7585  2.88 1.00   2.87    2.88 1.01 -0.80  6.56  7.35 0.06     0.01 0.01
y    3 8212 56.33 8.01  56.30   56.34 7.89 22.02 88.92 66.90 0.00     0.04 0.09
```

Again, we observe considerable bias in the means.

Let's examine the resulting missingness pattern to note something of interest here as well:

```
plot_a <- plot_pattern(mar_sim$amp)
plot_b <- plot_pattern(mnar_sim$amp)
plot_a + plot_b
```



*total number of missing entries: 6596 *total number of missing entries: 6599

Notice how the resulting patterns of missingness from imposing MAR and MNAR on our original data are very similar despite being the product of different mechanisms.

Conclusions

- The intuition behind MCAR, MAR and MNAR missingness mechanisms is relatively straightforward in theory, but in practice the missingness can take more or less complex forms and lead to varying implications for our understanding of the incomplete dataset.
- It is important not to forget that the missing data mechanisms are not the same as missing data patterns (Zhang 2023). The same missing data pattern may be consistent with often very different mechanisms, which suggests that we should be careful not to over-rely on the observed data to adjudicate between MAR and MNAR.
- Missing data mechanisms do not have unambiguous effects on the data or the conclusions we may draw from them. As Schouten and Vink (2021, 1254) argue "our results display various situations where the theoretical distinctions between missingness assumptions do not appropriately describe the actual effect of a missing data problem on statistical inference making". Thus, we may need to consider the underlying strength of correlations between the variables in the data, the shape of the function linking variable values to missingness, etc.
- And this is in simulations, the world of incomplete data out there is even messier plus we do not have the luxury of "knowing" the true values in the population nor how the missing data came about.

References

Buuren, Stef van. 2018. *Flexible Imputation of Missing Data, Second Edition.* Boca Raton, FL: CRC Press.

Enders, Craig K. 2022. *Applied Missing Data Analysis, Second Edition.* New York: The Guildford Press.

Schouten, Rianne Margaretha, Peter Lugtig, and Gerko Vink. 2018. "Generating Missing Values for Simulation Purposes: A Multivariate Amputation Procedure." *Journal of Statistical Computation and Simulation* 88 (15): 2909–30. https://doi.org/10.1080/00949655.2018.1491577.

Schouten, Rianne Margaretha, and Gerko Vink. 2021. "The Dance of the Mechanisms: How Observed Information Influences the Validity of Missingness Assumptions." *Sociological Methods & Research* 50 (3): 1243–58. https://doi.org/10.1177/0049124118799376.

Zhang, Xijuan. 2023. "How to Generate Missing Data For Simulation Studies." *The Quantitative Methods for Psychology* 19 (2): 100–122. https://doi.org/10.20982/tqmp.19.2.p100.