These are the five steps to ensuring missing data are correctly identified and appropriately dealt with:

1. Ensure your data are coded correctly.
2. Identify missing values within each variable.
3. Look for patterns of missingness.
4. Check for associations between missing and observed data.
5. Decide how to handle missing data.

Finalfit includes a number of functions to help with this.

**Some confusing terminology**

But first there are some terms which easy to mix up. These are important as they describe the mechanism of missingness and this determines how you can handle the missing data.

**Missing completely at random (MCAR)**

As it says, values are randomly missing from your dataset. Missing data values do not relate to any other data in the dataset and there is no pattern to the actual values of the missing data themselves.

For instance, when smoking status is not recorded in a random subset of patients.

This is easy to handle, but unfortunately, data are almost never missing completely at random.

**Missing at random (MAR)**

This is confusing and would be better stated as missing *conditionally* at random. Here, missing data *do* have a relationship with other variables in the dataset. However, the actual values that are missing are random.

For example, smoking status is not documented in female patients because the doctor was too shy to ask. Yes ok, not that realistic!

**Missing not at random (MNAR)**

The pattern of missingness is related to other variables in the dataset, but in addition, the values of the missing data are not random.

For example, when smoking status is not recorded in patients admitted as an emergency, who are also more likely to have worse outcomes from surgery.

Missing not at random data are important, can alter your conclusions, and are the most difficult to diagnose and handle. They can only be detected by collecting and examining some of the missing data. This is often difficult or impossible to do.

How you deal with missing data is dependent on the type of missingness. Once you know this, then you can sort it.

More on this below.

**1. Ensure your data are coded correctly: ff\_glimpse**

While clearly obvious, this step is often ignored in the rush to get results. The first step in any analysis is robust data cleaning and coding. Lots of packages have a glimpse function and finalfit is no different. This function has three specific goals:

1. Ensure all factors and numerics are correctly assigned. That is the commonest reason to get an error with a finalfit function. You think you’re using a factor variable, but in fact it is incorrectly coded as a continuous numeric.
2. Ensure you know which variables have missing data. This presumes missing values are correctly assigned NA.
3. Ensure factor levels and variable labels are assigned correctly.

**Example scenario**

Using the colon cancer dataset that comes with finalfit, we are interested in exploring the association between a cancer obstructing the bowel and 5-year survival, accounting for other patient and disease characteristics.

For demonstration purposes, we will create random MCAR and MAR smoking variables to the dataset.

# Make sure finalfit is up-to-date

install.packages("finalfit")

library(finalfit)

# Create some extra missing data

## Smoking missing completely at random

set.seed(1)

colon\_s$smoking\_mcar =

sample(c("Smoker", "Non-smoker", NA),

dim(colon\_s)[1], replace=TRUE,

prob = c(0.2, 0.7, 0.1)) %>%

factor()

Hmisc::label(colon\_s$smoking\_mcar) = "Smoking (MCAR)"

## Smoking missing conditional on patient sex

colon\_s$smoking\_mar[colon\_s$sex.factor == "Female"] =

sample(c("Smoker", "Non-smoker", NA),

sum(colon\_s$sex.factor == "Female"),

replace = TRUE,

prob = c(0.1, 0.5, 0.4))

colon\_s$smoking\_mar[colon\_s$sex.factor == "Male"] =

sample(c("Smoker", "Non-smoker", NA),

sum(colon\_s$sex.factor == "Male"),

replace=TRUE, prob = c(0.15, 0.75, 0.1))

colon\_s$smoking\_mar = factor(colon\_s$smoking\_mar)

Hmisc::label(colon\_s$smoking\_mar) = "Smoking (MAR)"

# Examine with ff\_glimpse

explanatory = c("age", "sex.factor",

"nodes", "obstruct.factor",

"smoking\_mcar", "smoking\_mar")

dependent = "mort\_5yr"

colon\_s %>%

ff\_glimpse(dependent, explanatory)

Numerics

label n missing\_n missing\_percent mean sd min max range se

age Age (years) 929 0 0.0 60 12 18 85 67 0.4

nodes NULL 911 18 1.9 4 4 0 33 33 0.1

Factors

label n missing\_n missing\_percent level\_n

sex.factor Sex 929 0 0 2

obstruct.factor Obstruction 908 21 2.3 2

mort\_5yr Mortality 5 year 915 14 1.5 2

smoking\_mcar Smoking (MCAR) 828 101 11 2

smoking\_mar Smoking (MAR) 719 210 23 2

levels levels\_count levels\_percent

sex.factor "Female", "Male" 445, 484 48, 52

obstruct.factor "No", "Yes" 732, 176, 21 78.8, 18.9, 2.3

mort\_5yr "Alive", "Died" 511, 404, 14 55.0, 43.5, 1.5

smoking\_mcar "Non-smoker", "Smoker" 645, 183, 101 69, 20, 11

smoking\_mar "Non-smoker", "Smoker" 591, 128, 210 64, 14, 23

The function summarises a data frame or tibble by numeric (continuous) variables and factor (discrete) variables. The dependent and explanatoryÂ  are for convenience. Pass either or neither e.g. to summarise data frame or tibble:

colon %>%

ff\_glimpse()

It doesn’t present well if you have factors with lots of levels, so you may want to remove these.

library(dplyr)

colon\_s %>%

select(-hospital) %>%

ff\_glimpse()

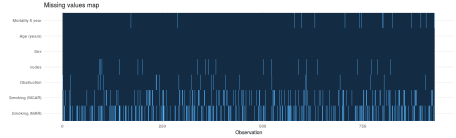
Use this to check that the variables are all assigned and behaving as expected. The proportion of missing data can be seen, e.g.Â smoking\_mar has 23% missing data.

**2. Identify missing values in each variable: missing\_plot**

In detecting patterns of missingness, this plot is useful. Row number is on the x-axis and all included variables are on the y-axis. Associations between missingness and observations can be easily seen, as can relationships of missingness between variables.

colon\_s %>%

missing\_plot()

[](https://i2.wp.com/www.datasurg.net/wp-content/uploads/2018/07/missing_values.png)

Click to enlarge.

It was only when writing this post that I discovered the amazing package, naniar. This package is recommended and provides lots of great visualisations for missing data.

Short Overview on naniar

# **A short overview of naniar**

Visualising missing data might sound a little strange - how do you visualise something that is not there? One approach to visualising missing data comes from ggobi and manet, which replaces NA values with values 10% lower than the minimum value in that variable. This visualisation is provided with the geom\_miss\_point() ggplot2 geom, which we illustrate by exploring the relationship between Ozone and Solar radiation from the airquality dataset.

library(ggplot2)

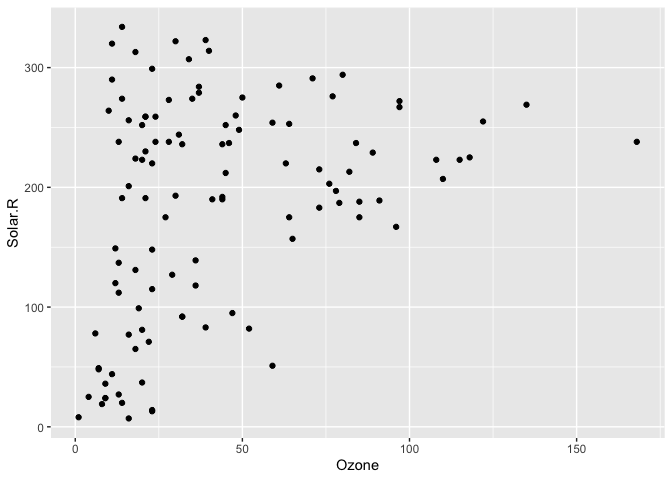
ggplot(data = airquality,

aes(x = Ozone,

y = Solar.R)) +

geom\_point()

#> Warning: Removed 42 rows containing missing values (geom\_point).



ggplot2 does not handle these missing values, and we get a warning message about the missing values.

We can instead use geom\_miss\_point() to display the missing data

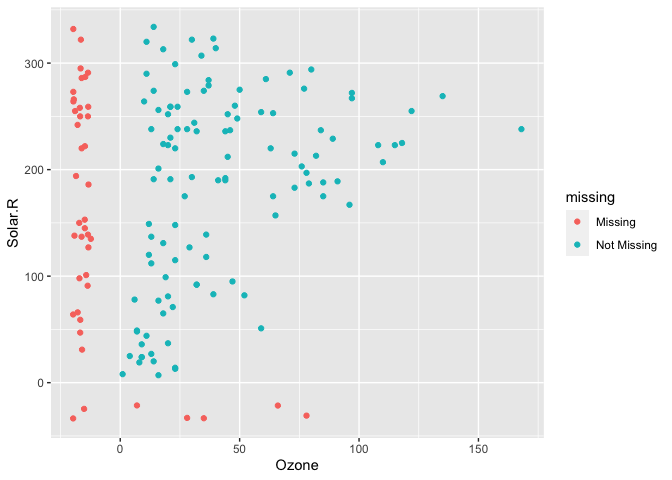
library(naniar)

ggplot(data = airquality,

aes(x = Ozone,

y = Solar.R)) +

geom\_miss\_point()



geom\_miss\_point() has shifted the missing values to now be 10% below the minimum value. The missing values are a different colour so that missingness becomes pre-attentive. As it is a ggplot2 geom, it supports features like faceting and other ggplot features.

p1 <-

ggplot(data = airquality,

aes(x = Ozone,

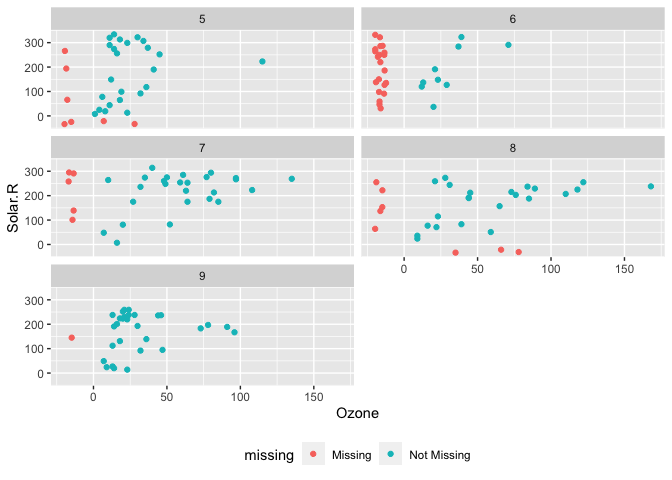
y = Solar.R)) +

geom\_miss\_point() +

facet\_wrap(~Month, ncol = 2) +

theme(legend.position = "bottom")

p1



# **Data Structures**

naniar provides a data structure for working with missing data, the shadow matrix (Swayne and Buja, 1998). The shadow matrix is the same dimension as the data, and consists of binary indicators of missingness of data values, where missing is represented as “NA”, and not missing is represented as “!NA”, and variable names are kep the same, with the added suffix “\_NA” to the variables.

head(airquality)

#> Ozone Solar.R Wind Temp Month Day

#> 1 41 190 7.4 67 5 1

#> 2 36 118 8.0 72 5 2

#> 3 12 149 12.6 74 5 3

#> 4 18 313 11.5 62 5 4

#> 5 NA NA 14.3 56 5 5

#> 6 28 NA 14.9 66 5 6

as\_shadow(airquality)

#> # A tibble: 153 x 6

#> Ozone\_NA Solar.R\_NA Wind\_NA Temp\_NA Month\_NA Day\_NA

#> <fct> <fct> <fct> <fct> <fct> <fct>

#> 1 !NA !NA !NA !NA !NA !NA

#> 2 !NA !NA !NA !NA !NA !NA

#> 3 !NA !NA !NA !NA !NA !NA

#> 4 !NA !NA !NA !NA !NA !NA

#> 5 NA NA !NA !NA !NA !NA

#> 6 !NA NA !NA !NA !NA !NA

#> 7 !NA !NA !NA !NA !NA !NA

#> 8 !NA !NA !NA !NA !NA !NA

#> 9 !NA !NA !NA !NA !NA !NA

#> 10 NA !NA !NA !NA !NA !NA

#> # … with 143 more rows

Binding the shadow data to the data you help keep better track of the missing values. This format is called “nabular”, a portmanteau of NA and tabular. You can bind the shadow to the data using bind\_shadow or nabular:

bind\_shadow(airquality)

#> # A tibble: 153 x 12

#> Ozone Solar.R Wind Temp Month Day Ozone\_NA Solar.R\_NA Wind\_NA Temp\_NA

#> <int> <int> <dbl> <int> <int> <int> <fct> <fct> <fct> <fct>

#> 1 41 190 7.4 67 5 1 !NA !NA !NA !NA

#> 2 36 118 8 72 5 2 !NA !NA !NA !NA

#> 3 12 149 12.6 74 5 3 !NA !NA !NA !NA

#> 4 18 313 11.5 62 5 4 !NA !NA !NA !NA

#> 5 NA NA 14.3 56 5 5 NA NA !NA !NA

#> 6 28 NA 14.9 66 5 6 !NA NA !NA !NA

#> 7 23 299 8.6 65 5 7 !NA !NA !NA !NA

#> 8 19 99 13.8 59 5 8 !NA !NA !NA !NA

#> 9 8 19 20.1 61 5 9 !NA !NA !NA !NA

#> 10 NA 194 8.6 69 5 10 NA !NA !NA !NA

#> # … with 143 more rows, and 2 more variables: Month\_NA <fct>, Day\_NA <fct>

nabular(airquality)

#> # A tibble: 153 x 12

#> Ozone Solar.R Wind Temp Month Day Ozone\_NA Solar.R\_NA Wind\_NA Temp\_NA

#> <int> <int> <dbl> <int> <int> <int> <fct> <fct> <fct> <fct>

#> 1 41 190 7.4 67 5 1 !NA !NA !NA !NA

#> 2 36 118 8 72 5 2 !NA !NA !NA !NA

#> 3 12 149 12.6 74 5 3 !NA !NA !NA !NA

#> 4 18 313 11.5 62 5 4 !NA !NA !NA !NA

#> 5 NA NA 14.3 56 5 5 NA NA !NA !NA

#> 6 28 NA 14.9 66 5 6 !NA NA !NA !NA

#> 7 23 299 8.6 65 5 7 !NA !NA !NA !NA

#> 8 19 99 13.8 59 5 8 !NA !NA !NA !NA

#> 9 8 19 20.1 61 5 9 !NA !NA !NA !NA

#> 10 NA 194 8.6 69 5 10 NA !NA !NA !NA

#> # … with 143 more rows, and 2 more variables: Month\_NA <fct>, Day\_NA <fct>

Using the nabular format helps you manage where missing values are in your dataset and make it easy to do visualisations where you split by missingness:

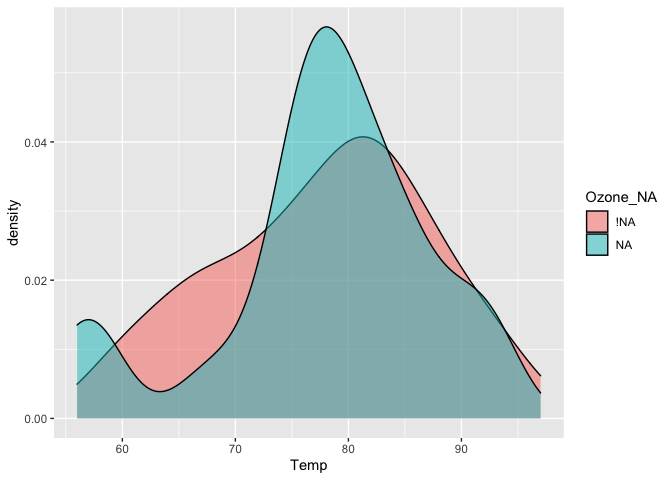
airquality %>%

bind\_shadow() %>%

ggplot(aes(x = Temp,

fill = Ozone\_NA)) +

geom\_density(alpha = 0.5)



And even visualise imputations

airquality %>%

bind\_shadow() %>%

as.data.frame() %>%

simputation::impute\_lm(Ozone ~ Temp + Solar.R) %>%

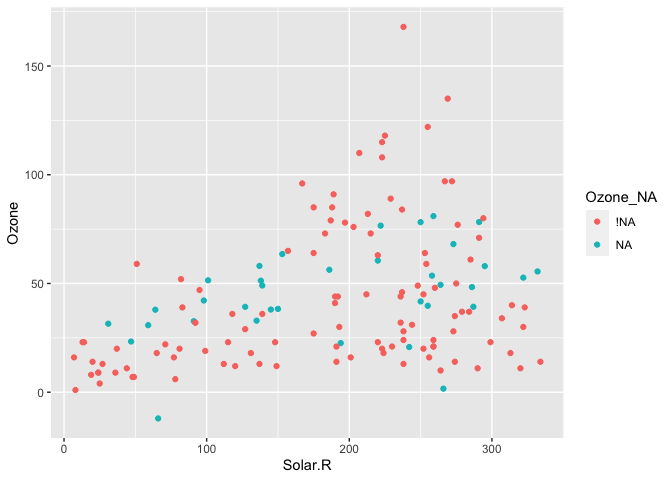
ggplot(aes(x = Solar.R,

y = Ozone,

colour = Ozone\_NA)) +

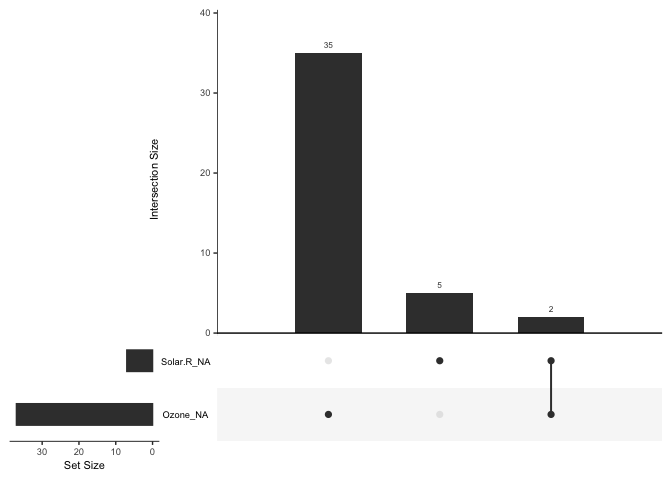
geom\_point()

#> Warning: Removed 7 rows containing missing values (geom\_point).



Or perform an upset plot - to plot of the combinations of missingness across cases, using the gg\_miss\_upset function

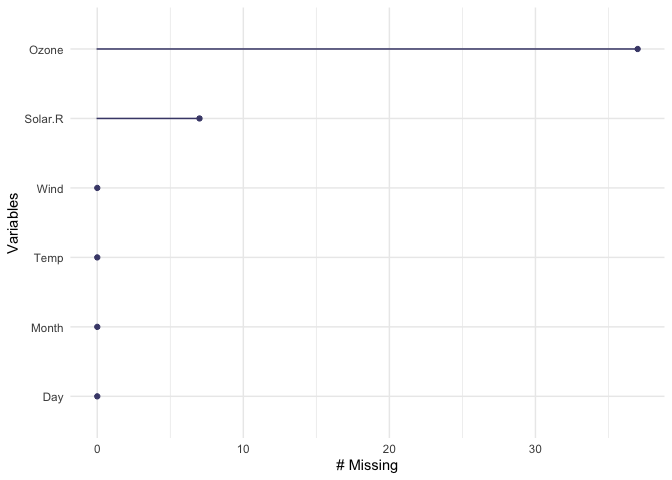
gg\_miss\_upset(airquality)



naniar does this while following consistent principles that are easy to read, thanks to the tools of the tidyverse.

naniar also provides handy visualations for each variable:

gg\_miss\_var(airquality)

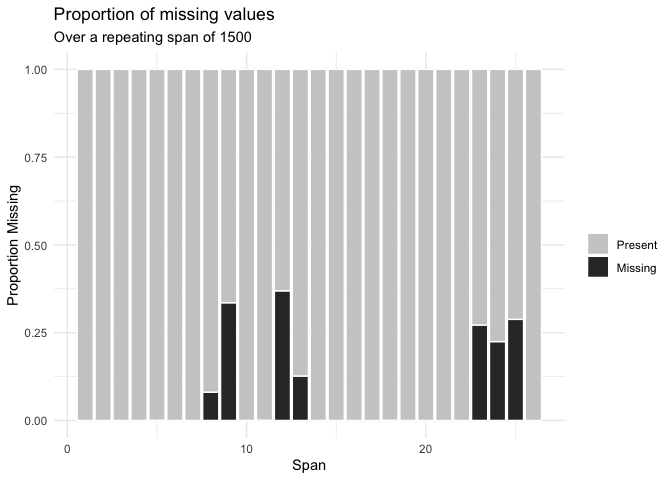


Or the number of missings in a given variable at a repeating span

gg\_miss\_span(pedestrian,

var = hourly\_counts,

span\_every = 1500)



You can read about all of the visualisations in naniar in the vignette Gallery of missing data visualisations using naniar.

naniar also provides handy helpers for calculating the number, proportion, and percentage of missing and complete observations:

n\_miss(airquality)

#> [1] 44

n\_complete(airquality)

#> [1] 874

prop\_miss(airquality)

#> [1] 0.04793028

prop\_complete(airquality)

#> [1] 0.9520697

pct\_miss(airquality)

#> [1] 4.793028

pct\_complete(airquality)

#> [1] 95.20697

# **Numerical summaries for missing data**

naniar provides numerical summaries of missing data, that follow a consistent rule that uses a syntax begining with miss\_. Summaries focussing on variables or a single selected variable, start with miss\_var\_, and summaries for cases (the initial collected row order of the data), they start with miss\_case\_. All of these functions that return dataframes also work with dplyr’s group\_by().

For example, we can look at the number and percent of missings in each case and variable with miss\_var\_summary(), and miss\_case\_summary(), which both return output ordered by the number of missing values.

miss\_var\_summary(airquality)

#> # A tibble: 6 x 3

#> variable n\_miss pct\_miss

#> <chr> <int> <dbl>

#> 1 Ozone 37 24.2

#> 2 Solar.R 7 4.58

#> 3 Wind 0 0

#> 4 Temp 0 0

#> 5 Month 0 0

#> 6 Day 0 0

miss\_case\_summary(airquality)

#> # A tibble: 153 x 3

#> case n\_miss pct\_miss

#> <int> <int> <dbl>

#> 1 5 2 33.3

#> 2 27 2 33.3

#> 3 6 1 16.7

#> 4 10 1 16.7

#> 5 11 1 16.7

#> 6 25 1 16.7

#> 7 26 1 16.7

#> 8 32 1 16.7

#> 9 33 1 16.7

#> 10 34 1 16.7

#> # … with 143 more rows

You could also group\_by() to work out the number of missings in each variable across the levels within it.

library(dplyr)

#>

#> Attaching package: 'dplyr'

#> The following objects are masked from 'package:stats':

#>

#> filter, lag

#> The following objects are masked from 'package:base':

#>

#> intersect, setdiff, setequal, union

airquality %>%

group\_by(Month) %>%

miss\_var\_summary()

#> # A tibble: 25 x 4

#> # Groups: Month [5]

#> Month variable n\_miss pct\_miss

#> <int> <chr> <int> <dbl>

#> 1 5 Ozone 5 16.1

#> 2 5 Solar.R 4 12.9

#> 3 5 Wind 0 0

#> 4 5 Temp 0 0

#> 5 5 Day 0 0

#> 6 6 Ozone 21 70

#> 7 6 Solar.R 0 0

#> 8 6 Wind 0 0

#> 9 6 Temp 0 0

#> 10 6 Day 0 0

#> # … with 15 more rows

You can read more about all of these functions in the vignette “Getting Started with naniar”.

# **Statistical tests of missingness**

naniar provides mcar\_test() for Little’s (1988) statistical test for missing completely at random (MCAR) data. The null hypothesis in this test is that the data is MCAR, and the test statistic is a chi-squared value. Given the high statistic value and low p-value, we can conclude that the airquality data is not missing completely at random:

mcar\_test(airquality)

#> # A tibble: 1 x 4

#> statistic df p.value missing.patterns

#> <dbl> <dbl> <dbl> <int>

#> 1 35.1 14 0.00142 4

**3. Look for patterns of missingness: missing\_pattern**

missing\_pattern simply wraps mice::md.pattern using finalfit grammar. This produces a table and a plot showing the pattern of missingness between variables.

explanatory = c("age", "sex.factor",

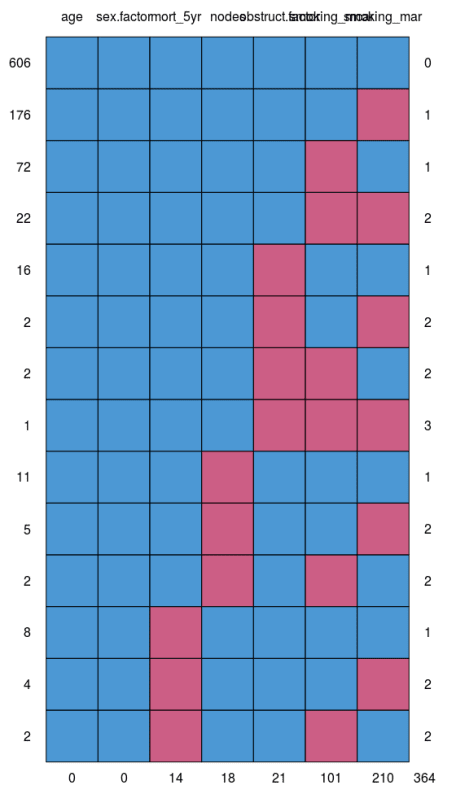
"nodes", "obstruct.factor",

"smoking\_mcar", "smoking\_mar")

dependent = "mort\_5yr"

colon\_s %>%

missing\_pattern(dependent, explanatory)

[](https://i0.wp.com/www.datasurg.net/wp-content/uploads/2018/07/missing_pattern.png)

This allows us to look for patterns of missingness *between* variables. There are 14 patterns in this data. The number and pattern of missingness help us to determine the likelihood of it being random rather than systematic.Â

**Make sure you include missing data in demographics tables**

Table 1 in a healthcare study is often a demographics table of an “explanatory variable of interest” against other explanatory variables/confounders. Do not silently drop missing values in this table. It is easy to do this correctly with summary\_factorlist. This function provides a useful summary of a dependent variable against explanatory variables. Despite its name, continuous variables are handled nicely.

na\_include=TRUE ensures missing data from the explanatory variables (but not dependent) are included. Note that any p-values are generated across missing groups as well, so run a second time with na\_include=FALSE if you wish a hypothesis test only over observed data.

library(finalfit)

# Explanatory or confounding variables

explanatory = c("age", "sex.factor",

"nodes",

"smoking\_mcar", "smoking\_mar")

# Explanatory variable of interest

dependent = "obstruct.factor" # Bowel obstruction

colon\_s %>%

summary\_factorlist(dependent, explanatory,

na\_include=TRUE, p=TRUE)

label levels No Yes p

Age (years) Mean (SD) 60.2 (11.5) 57.3 (13.3) 0.014

Sex Female 346 (79.2) 91 (20.8) 0.290

Male 386 (82.0) 85 (18.0)

nodes Mean (SD) 3.7 (3.7) 3.5 (3.2) 0.774

Smoking (MCAR) Non-smoker 500 (79.4) 130 (20.6) 0.173

Smoker 154 (85.6) 26 (14.4)

Missing 78 (79.6) 20 (20.4)

Smoking (MAR) Non-smoker 467 (80.9) 110 (19.1) 0.056

Smoker 91 (73.4) 33 (26.6)

Missing 174 (84.1) 33 (15.9)

**4. Check for associations between missing and observed data: missing\_pairs | missing\_compare**

In deciding whether data is MCAR or MAR, one approach is to explore patterns of missingness between levels of included variables. This is particularly important (I would say absolutely required) for a primary outcome measure / dependent variable.

Take for example “death”. When that outcome is missing it is often for a particular reason. For example, perhaps patients undergoing emergency surgery were less likely to have complete records compared with those undergoing planned surgery. And of course, death is more likely after emergency surgery.

missing\_pairs uses functions from the excellent GGally package. It produces pairs plots to show relationships between missing values and observed values in all variables.

explanatory = c("age", "sex.factor",

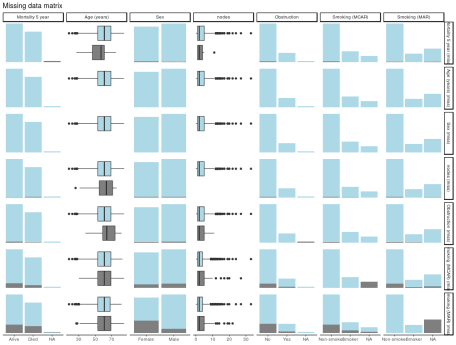
"nodes", "obstruct.factor",

"smoking\_mcar", "smoking\_mar")

dependent = "mort\_5yr"

colon\_s %>%

missing\_pairs(dependent, explanatory)

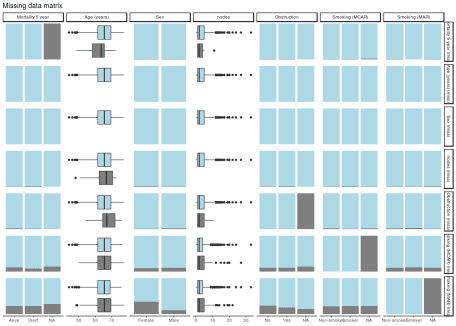
[](https://i1.wp.com/www.datasurg.net/wp-content/uploads/2018/07/missing_pairs1.png)

For continuous variables (age and nodes), the distributions of observed and missing data can be visually compared. Is there a difference between age and mortality above?

For discrete, data, counts are presented by default. It is often easier to compare proportions:

colon\_s %>%

missing\_pairs(dependent, explanatory, position = "fill", )

[](https://i1.wp.com/www.datasurg.net/wp-content/uploads/2018/07/missing_pairs2-1.png)

It should be obvious that missingness in Smoking (MCAR) does not relate to sex (row 6, column 3). But missingnessÂ  in Smoking (MAR) does differ by sex (last row, column 3) as was designed above when the missing data were created.

We can confirm this using missing\_compare.

explanatory = c("age", "sex.factor",

"nodes", "obstruct.factor")

dependent = "smoking\_mcar"

colon\_s %>%

missing\_compare(dependent, explanatory)

Missing data analysis: Smoking (MCAR) Not missing Missing p

Age (years) Mean (SD) 59.7 (11.9) 59.9 (12.6) 0.867

Sex Female 399 (89.7) 46 (10.3) 0.616

Male 429 (88.6) 55 (11.4)

nodes Mean (SD) 3.6 (3.4) 4 (4.5) 0.990

Obstruction No 654 (89.3) 78 (10.7) 0.786

Yes 156 (88.6) 20 (11.4)

dependent = "smoking\_mar"

colon\_s %>%

missing\_compare(dependent, explanatory)

Missing data analysis: Smoking (MAR) Not missing Missing p

Age (years) Mean (SD) 59.6 (11.9) 60.1 (12) 0.709

Sex Female 288 (64.7) 157 (35.3) <0.001

Male 431 (89.0) 53 (11.0)

nodes Mean (SD) 3.6 (3.6) 3.8 (3.6) 0.730

Obstruction No 558 (76.2) 174 (23.8) 0.154

Yes 143 (81.2) 33 (18.8)

It takes “dependent” and “explanatory” variables, but in this context “dependent” just refers to the variable being tested for missingness against the “explanatory” variables.

Comparisons for continuous data use a Kruskal Wallis and for discrete data a chi-squared test.

As expected, a relationship is seen between Sex and Smoking (MAR) but not Smoking (MCAR).

**For those who like an omnibus test**

If you are work predominately with numeric rather than discrete data (categorical/factors), you may find these tests from the MissMech package useful..

library(finalfit)

library(dplyr)

library(MissMech)

explanatory = c("age", "nodes")

dependent = "mort\_5yr"

colon\_s %>%

select(explanatory) %>%

MissMech::TestMCARNormality()

**5. Decide how to handle missing data**

Prior to a standard regression analysis, we can either:

* Delete the variable with the missing data
* Delete the cases with the missing data
* Impute (fill in) the missing data
* Model the missing data

**MCAR, MAR, or MNAR**

**MCAR vs MAR**

Using the examples, we identify that Smoking (MCAR) is missing completely at random.Â

We know nothing about the missing values themselves, but we know of no plausible reason that the values of the missing data, for say, people who died should be different to the values of the missing data for those who survived. The pattern of missingness is therefore not felt to be MNAR.

**Common solution**

Depending on the number of data points that are missing, we may have sufficient power with complete cases to examine the relationships of interest.

We therefore elect to simply omit the patients in whom smoking is missing. This is known as list-wise deletion and will be performed by default in standard regression analyses including finalfit.

explanatory = c("age", "sex.factor",

"nodes", "obstruct.factor",

"smoking\_mcar")

dependent = "mort\_5yr"

colon\_s %>%

finalfit(dependent, explanatory, metrics=TRUE)

Dependent: Mortality 5 year Alive Died OR (univariable) OR (multivariable)

Age (years) Mean (SD) 59.8 (11.4) 59.9 (12.5) 1.00 (0.99-1.01, p=0.986) 1.01 (1.00-1.02, p=0.200)

Sex Female 243 (47.6) 194 (48.0) - -

Male 268 (52.4) 210 (52.0) 0.98 (0.76-1.27, p=0.889) 1.02 (0.76-1.38, p=0.872)

nodes Mean (SD) 2.7 (2.4) 4.9 (4.4) 1.24 (1.18-1.30, p<0.001) 1.25 (1.18-1.33, p<0.001)

Obstruction No 408 (82.1) 312 (78.6) - -

Yes 89 (17.9) 85 (21.4) 1.25 (0.90-1.74, p=0.189) 1.53 (1.05-2.22, p=0.027)

Smoking (MCAR) Non-smoker 358 (79.9) 277 (75.3) - -

Smoker 90 (20.1) 91 (24.7) 1.31 (0.94-1.82, p=0.113) 1.37 (0.96-1.96, p=0.083)

"Number in dataframe = 929, Number in model = 782, Missing = 147, AIC = 1003.3, C-statistic = 0.687, H&L = Chi-sq(8) 15.03 (p=0.058)

**Other considerations**

1. Sensitivity analysis
2. Omit the variable
3. Imputation
4. Model the missing data

If the variable in question is thought to be particularly important, you may wish to perform a sensitivity analysis. A sensitivity analysis in this context aims to capture the effect of uncertainty on the conclusions drawn from the model. Thus, you may choose to re-label all missing smoking values as “smoker”, and see if that changes the conclusions of your analysis. The same procedure can be performed labeling with “non-smoker”.

If smoking is not associated with the explanatory variable of interest (bowel obstruction) or the outcome, it may be considered not to be a confounderÂ  and so could be omitted. That neatly deals with the missing data issue, but of course may not be appropriate.

Imputation and modelling are considered below.

**MCAR vs MAR**

But life is rarely that simple.

Consider that the smoking variable is more likely to be missing if the patient is female (missing\_compareshows a relationship). But, say, that the missing values are not different from the observed values. Missingness is then MAR.

If we simply drop all the cases (patients) in which smoking is missing (list-wise deletion), then proportionality we drop more females than men. This may have consequences for our conclusions if sex is associated with our explanatory variable of interest or outcome.

**Common solution**

mice is our go to package for multiple imputation. That’s the process of filling in missing data using a best-estimate from all the other data that exists. When first encountered, this doesn’t sounds like a good idea.

However, taking our simple example, if missingness in smoking is predicted strongly by sex, and the values of the missing data are random, then we can impute (best-guess) the missing smoking values using sex and other variables in the dataset.

Imputation is not usually appropriate for the explanatory variable of interest or the outcome variable. With both of these, the hypothesis is that there is an meaningful association with other variables in the dataset, therefore it doesn’t make sense to use these variables to impute them.

Here is some code to run mice. The package is well documented, and there are a number of checks and considerations that should be made to inform the imputation process. Read the documentation carefully prior to doing this yourself.

# Multivariate Imputation by Chained Equations (mice)

library(finalfit)

library(dplyr)

library(mice)

explanatory = c("age", "sex.factor",

"nodes", "obstruct.factor", "smoking\_mar")

dependent = "mort\_5yr"

colon\_s %>%

select(dependent, explanatory) %>%

# Exclude outcome and explanatory variable of interest from imputation

dplyr::filter(!is.na(mort\_5yr), !is.na(obstruct.factor)) %>%

# Run imputation with 10 imputed sets

mice(m = 10) %>%

# Run logistic regression on each imputed set

with(glm(formula(ff\_formula(dependent, explanatory)),

family="binomial")) %>%

# Pool and summarise results

pool() %>%

summary(conf.int = TRUE, exponentiate = TRUE) %>%

# Jiggle into finalfit format

mutate(explanatory\_name = rownames(.)) %>%

select(explanatory\_name, estimate, `2.5 %`, `97.5 %`, p.value) %>%

condense\_fit(estimate\_suffix = " (multiple imputation)") %>%

remove\_intercept() -> fit\_imputed

# Use finalfit merge methods to create and compare results

colon\_s %>%

summary\_factorlist(dependent, explanatory, fit\_id = TRUE) -> summary1

colon\_s %>%

glmuni(dependent, explanatory) %>%

fit2df(estimate\_suffix = " (univariable)") -> fit\_uni

colon\_s %>%

glmmulti(dependent, explanatory) %>%

fit2df(estimate\_suffix = " (multivariable inc. smoking)") -> fit\_multi

explanatory = c("age", "sex.factor",

"nodes", "obstruct.factor")

colon\_s %>%

glmmulti(dependent, explanatory) %>%

fit2df(estimate\_suffix = " (multivariable)") -> fit\_multi\_r

# Combine to final table

summary1 %>%

ff\_merge(fit\_uni) %>%

ff\_merge(fit\_multi\_r) %>%

ff\_merge(fit\_multi) %>%

ff\_merge(fit\_imputed) %>%

select(-fit\_id, -index)

label levels Alive Died OR (univariable) OR (multivariable) OR (multivariable inc. smoking) OR (multiple imputation)

Age (years) Mean (SD) 59.8 (11.4) 59.9 (12.5) 1.00 (0.99-1.01, p=0.986) 1.01 (1.00-1.02, p=0.122) 1.02 (1.00-1.03, p=0.010) 1.01 (1.00-1.02, p=0.116)

Sex Female 243 (55.6) 194 (44.4) - - - -

Male 268 (56.1) 210 (43.9) 0.98 (0.76-1.27, p=0.889) 0.98 (0.74-1.30, p=0.890) 0.88 (0.64-1.23, p=0.461) 0.99 (0.75-1.31, p=0.957)

nodes Mean (SD) 2.7 (2.4) 4.9 (4.4) 1.24 (1.18-1.30, p<0.001) 1.25 (1.19-1.32, p<0.001) 1.25 (1.18-1.33, p<0.001) 1.25 (1.19-1.32, p<0.001)

Obstruction No 408 (56.7) 312 (43.3) - - - -

Yes 89 (51.1) 85 (48.9) 1.25 (0.90-1.74, p=0.189) 1.36 (0.95-1.93, p=0.089) 1.26 (0.85-1.88, p=0.252) 1.36 (0.95-1.93, p=0.089)

Smoking (MAR) Non-smoker 328 (56.4) 254 (43.6) - - - -

Smoker 68 (53.5) 59 (46.5) 1.12 (0.76-1.65, p=0.563) - 1.25 (0.82-1.89, p=0.300) 1.26 (0.82-1.94, p=0.289)

By examining the coefficients, the effect of the imputation compared with the complete case analysis can be clearly seen.

**Other considerations**

1. Omit the variable
2. Imputing factors with new level for missing data
3. Model the missing data

As above, if the variable does not appear to be important, it may be omitted from the analysis. A sensitivity analysis in this context is another form of imputation. But rather than using all other available information to best-guess the missing data, we simply assign the value as above. Imputation is therefore likely to be more appropriate.

There is an alternative method to model the missing data for the categorical in this setting – just consider the missing data as a factor level. This has the advantage of simplicity, with the disadvantage of increasing the number of terms in the model. Multiple imputation is generally preferred.Â

library(dplyr)

colon\_s %>%

mutate(

smoking\_mar = forcats::fct\_explicit\_na(smoking\_mar)

) %>%

finalfit(dependent, explanatory)

Dependent: Mortality 5 year Alive Died OR (univariable) OR (multivariable)

Age (years) Mean (SD) 59.8 (11.4) 59.9 (12.5) 1.00 (0.99-1.01, p=0.986) 1.01 (1.00-1.02, p=0.119)

Sex Female 243 (47.6) 194 (48.0) - -

Male 268 (52.4) 210 (52.0) 0.98 (0.76-1.27, p=0.889) 0.96 (0.72-1.30, p=0.809)

nodes Mean (SD) 2.7 (2.4) 4.9 (4.4) 1.24 (1.18-1.30, p<0.001) 1.25 (1.19-1.32, p<0.001)

Obstruction No 408 (82.1) 312 (78.6) - -

Yes 89 (17.9) 85 (21.4) 1.25 (0.90-1.74, p=0.189) 1.34 (0.94-1.91, p=0.102)

Smoking (MAR) Non-smoker 328 (64.2) 254 (62.9) - -

Smoker 68 (13.3) 59 (14.6) 1.12 (0.76-1.65, p=0.563) 1.24 (0.82-1.88, p=0.308)

(Missing) 115 (22.5) 91 (22.5) 1.02 (0.74-1.41, p=0.895) 0.99 (0.69-1.41, p=0.943)

**MNAR vs MAR**

Missing not at random data is tough in healthcare. To determine if data are MNAR for definite, we need to know their value in a subset of observations (patients).

Using our example above. Say smoking status is poorly recorded in patients admitted to hospital as an emergency with an obstructing cancer. Obstructing bowel cancers may be larger or their position may make the prognosis worse. Smoking may relate to the aggressiveness of the cancer and may be an independent predictor of prognosis. The missing values for smoking may therefore not random. Smoking may be more common in the emergency patients and may be more common in those that die.

There is no easy way to handle this. If at all possible, try to get the missing data. Otherwise, take care when drawing conclusions from analyses where data are thought to be missing not at random.Â