Data import & descriptives

Magnus Johansson

2023-06-12

Overview of this file

- importing data
- looking at data
- descriptives
  - tables
  - figures
  - some basic statistical analyses
- missing data
- wrangling data (wide to long, etc)

Data cleaning

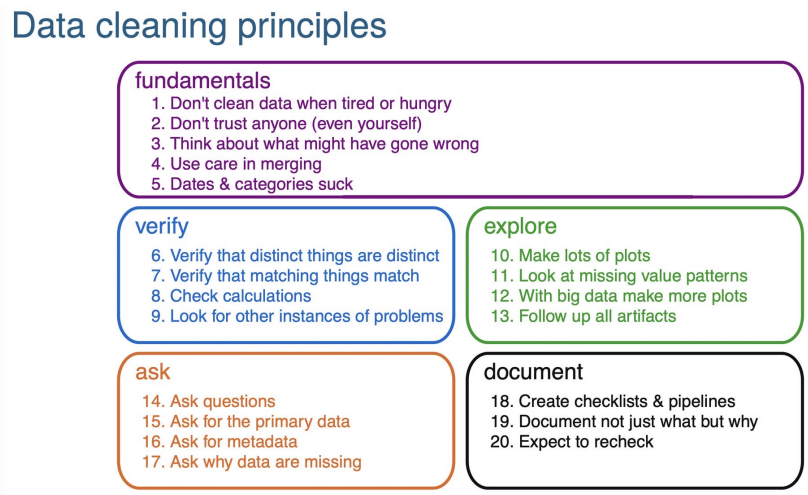


Figure 1: Image credit: https://kbroman.org/Talk_DataCleaning/data_cleaning.pdf

Setting up for data analysis

Let's load packages/libraries.

```
# these are mostly for data management/wrangling and visualization
library(tidyverse) # for most things
library(foreign) # for reading SPSS files
library(readxl) # read MS Excel files
library(showtext) # get fonts
library(glue) # simplifies mixing text and code in figures and tables
library(arrow) # support for efficient file formats
library(grateful) # create table+references for packages used in a project
library(styler) # only a one-time installation (it is an Rstudio plugin)
library(car) # for car::recode only
library(skimr) # data skimming
library(lubridate) # for handling dates in data
library(janitor) # for many things in data cleaning

# these are mostly for data analysis and visualization
library(gtsummary)
library(scales)
library(visdat)
library(psych)
library(lme4)
library(nlme)
library(broom.mixed)
library(ggplot2)
library(patchwork)
library(easystats)
library(GGally)
library(mice)
library(modelsummary)
library(ggside)
library(ggdist)
library(kableExtra)
library(formattable)
library(ggrepel)
```

Define a ggplot theme `theme_ki()` and standard table function, `kbl_ki()`.

```
source("ki.R") # this reads an external file and loads whatever is in it
```

Adaptions

Some functions exist in multiple packages, which can be a source of headaches and confusion.

Below we define preferred functions that are frequently used. If desired, we can still use specific functions by using their package prefix, for instance `dplyr::recode()`.

```
#library(conflicted)
select <- dplyr::select
count <- dplyr::count
recode <- car::recode
rename <- dplyr::rename
filter <- dplyr::filter
clean_names <- janitor::clean_names
```

Importing data

The open dataset we will use for our experiments was retrieved from https://doi.org/10.26180/13240304 and is available in the `data` subfolder of the R project folder we are currently working in. The description of the dataset on Figshare is:

> De-identified dataset from a randomised controlled trial of Mindfulness-integrated cognitive behaviour therapy (MiCBT) versus a treatment-as-usual waitlist control. All participants completed the measures one week before the start of the MiCBT group intervention (T0), after week 4 (T1), at week 8 (T2, post-intervention), and then again after a 6-month follow up period (T3). A full description of the project methodology including the measures used in the trial is provided in the protocol paper (see References).

And from the study protocol:

> The intent of this study is to examine the effectiveness of MiCBT to create changes in clinical measures of depression, anxiety and stress. It is hypothesized that these changes will occur during the program in stages 1,2 and 3 and be enhanced in stage 4 because of the additional practice time. Compassion and ethics are taught in Stage 4 for relapse prevention which is not the focus of the current study.

Looking at the abbreviations section of the study protocol, we can hopefully get some variable name explanations:

- EQ: Experiences Questionnaire
- FS: Flourishing scale
- K10: Kessler Psychological Distress Scale
- MAIS: Multidimensional Assessment of Interoceptive Awareness
- MB-EAT: Mindfulness-based Eating Program
- MBI: Mindfulness-based Intervention
- MBRE: Mindfulness-based Relationship Enhancement
- MBSR: Mindfulness-based Stress Reducyion
- MiCBT: Mindfulness integrated Cognitive Behavior Therapy
- MSES: Mindfulness-based Self-efficacy Scale
- NAS: Non-attachment Scale
- SWLS: Satisfaction with Life Scale

Let's read the datafile and have a first look.

```
df <- read_excel("data/MiCBT RCT data_Bridges repository.xlsx")
```

Look in the Environment quadrant (upper right). How many observations and variables do we have in the `df` object?

Press the circle to the left of `df` to get a quick look at the data. We can see the word "missing" noted in several fields. Anything else you notice about the variables?

Let's re-import the data and tell `read_excel()` to code missing correctly.

```
df <- read_excel("data/MiCBT RCT data_Bridges repository.xlsx",
                 na = "missing")
```

Have another look at the data now and see what happened. You can go back and run the previous chunk to see the difference more clearly.

Also, have a look at the naming scheme and see what pattern you find?

The K10 questionnaire is used for pre-intervention measurement and screening, as well as follow-up measurement. Let's look at the variables containing "K10".

```
df %>%
  select(contains("K10"))
```

```
# A tibble: 106 x 7
   K10_Score_GP GPK10_coded TK10_t0 K10_di_t0    TK10_t1 TK10_t2 TK10_t3
          <dbl>       <dbl>   <dbl> <chr>          <dbl>   <dbl>   <dbl>
 1           33           1      30 30+               25      30      21
 2           24           0      30 30+               27      31      32
 3           36           1      38 30+               28      18      23
 4           29           0      27 less than 30      21      24      27
 5           27           0      22 less than 30      32      NA      23
 6           29           0      16 less than 30      15      16      17
 7           23           0      23 less than 30      20      15      17
 8           27           0      30 30+               42      27      19
 9           29           0      25 less than 30      17      16      15
10           30           1      29 less than 30      21      17      21
# i 96 more rows
```

`K10_di_t0` is a categorical variable created from `TK10_t0`, and it does not repeat for other time points. As such, it is mislabeled and we want to fix this. While we are at it, we can rename some other variables too.

The syntax for `dplyr::rename()` is `newname = oldname`.

```
df <- df %>%
  rename(id = BridgesID,
         Group = GROUP,
         K10preCat = K10_di_t0)
```

Demographics

The dataset does not include any demographics. Just for fun, we'll add randomly assigned age and gender variables. `mutate()` helps us create or modify variables.

```
df <- df %>%
  mutate(age = rnorm(nrow(df),
                     mean = 44,
                     sd = 8),
         age = as.integer(age),
         age = recode(age,"0:18=19"),
         sex = sample(1:2, nrow(df), replace=TRUE))
```

```
summary(df$age)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  26.00   38.25   43.00   43.32   48.00   67.00
```

`summary()` is a general function that can be used with many types of objects, including model outputs.

Tables

All participants

```
df %>%
  select(age) %>%
  tbl_summary()
```

Characteristic	**N = 106**
age	43 (38, 48)

Control/intervention group

```
df %>%
  select(age,Group) %>%
  tbl_summary(by = Group,
              statistic = list(all_continuous() ~ "{mean} ({sd}, {min}-{max})"))
```

Characteristic	**Control**, N = 55	**MiCBT**, N = 51
age	45 (8, 29-67)	41 (7, 26-56)

Difference?

We can see that there isn't a difference, and we expect this since random sampling was used. But if you wanted to test the difference, this is one way.

```
age.test <- t.test(age ~ Group, data = df)
age.test
```

```
    Welch Two Sample t-test

data:  age by Group
t = 2.6752, df = 103.22, p-value = 0.008685
alternative hypothesis: true difference in means between group Control and group MiCBT is
95 percent confidence interval:
 1.029905 6.933018
sample estimates:
mean in group Control   mean in group MiCBT
             45.23636              41.25490
```

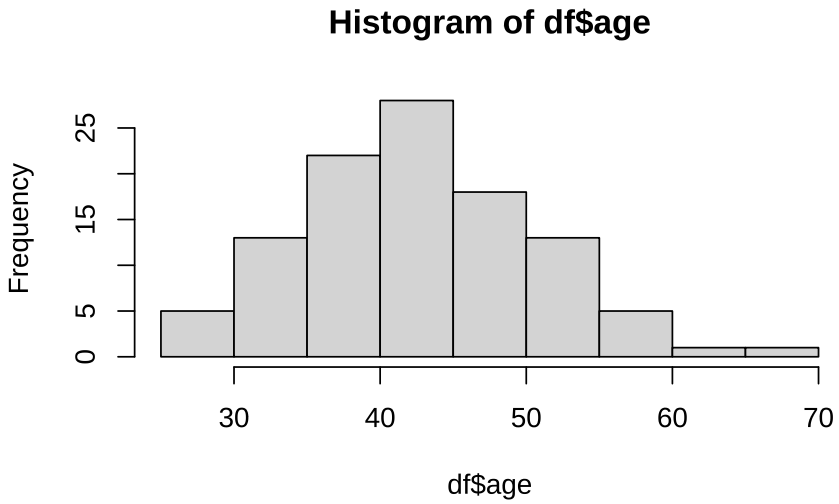
```r
tidy(age.test)
```

```
# A tibble: 1 x 10
  estimate estimate1 estimate2 statistic p.value parameter conf.low conf.high
     <dbl>     <dbl>     <dbl>     <dbl>   <dbl>     <dbl>    <dbl>     <dbl>
1     3.98      45.2      41.3      2.68 0.00869      103.     1.03      6.93
# i 2 more variables: method <chr>, alternative <chr>
```

Figures

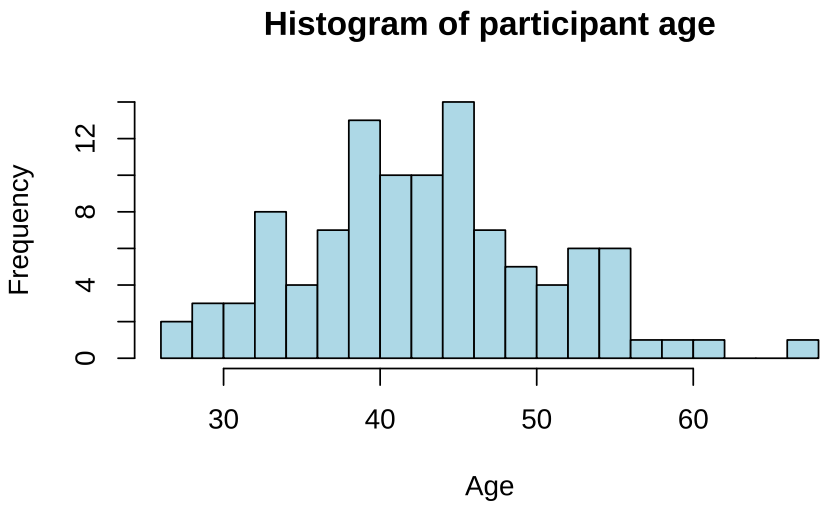
Base R examples.

```r
hist(df$age)
```



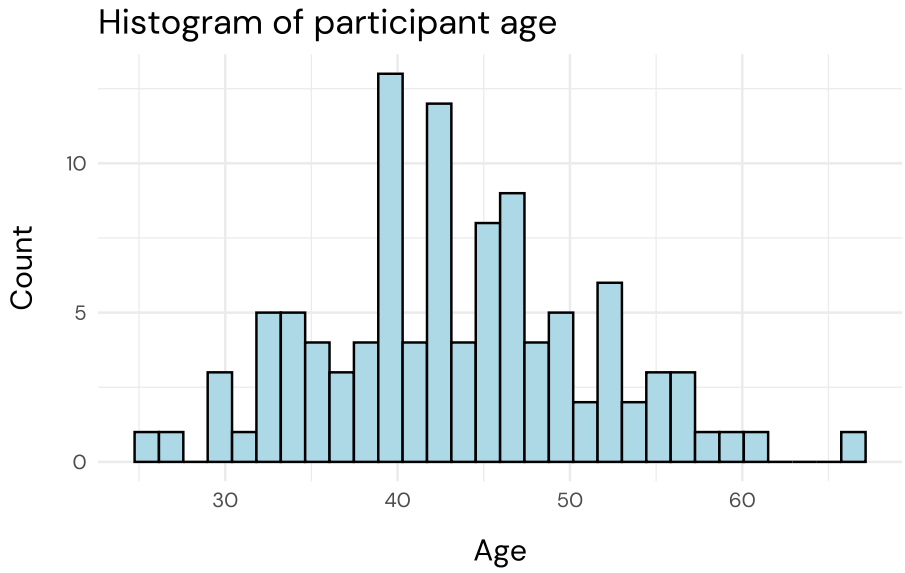

```r
hist(df$age,
     col = "lightblue",
```

```
    main = "Histogram of participant age",
    xlab = "Age",
    breaks = 24)
```



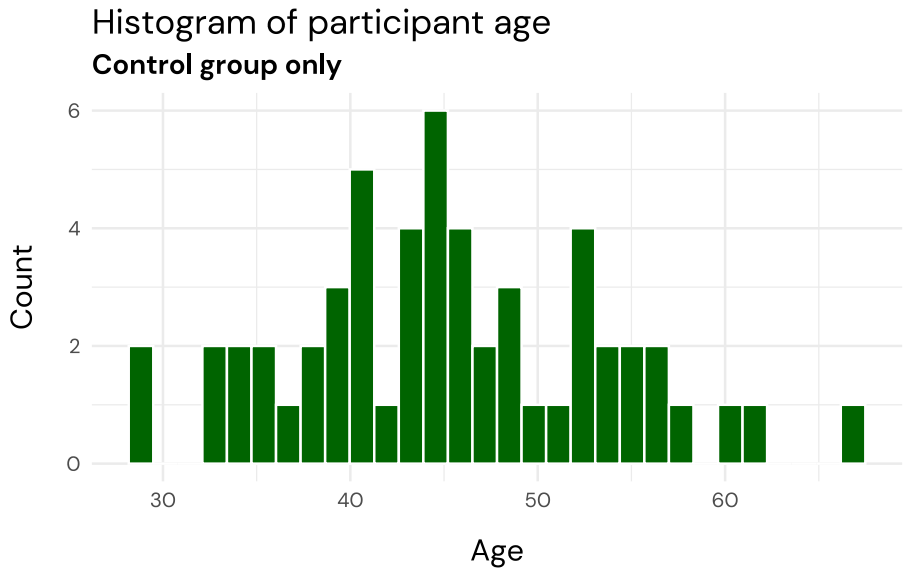
With ggplot we have a lot more flexibility. Note that as soon as `ggplot()` has been called, the line ends with `+` when we add plot configurations.

```
df %>%
  ggplot(aes(x = age)) +
  geom_histogram(fill = "lightblue",
                 color = "black") +
  labs(title = "Histogram of participant age",
       x = "Age",
       y = "Count") +
  theme_ki()
```



Let's look separately at the Control group.

```
df %>%
  filter(Group == "Control") %>%
  ggplot(aes(x = age)) +
  geom_histogram(fill = "darkgreen",
                 color = "white") +
  labs(title = "Histogram of participant age",
       x = "Age",
       y = "Count",
       subtitle = "Control group only") +
  theme_ki()
```

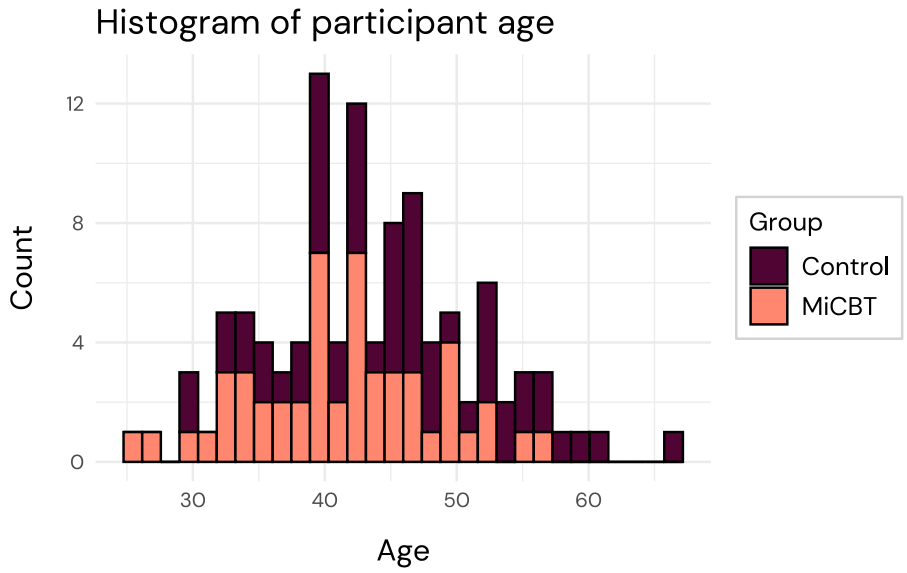


Fill/color based on Group variable.

We can have both in the same plot.

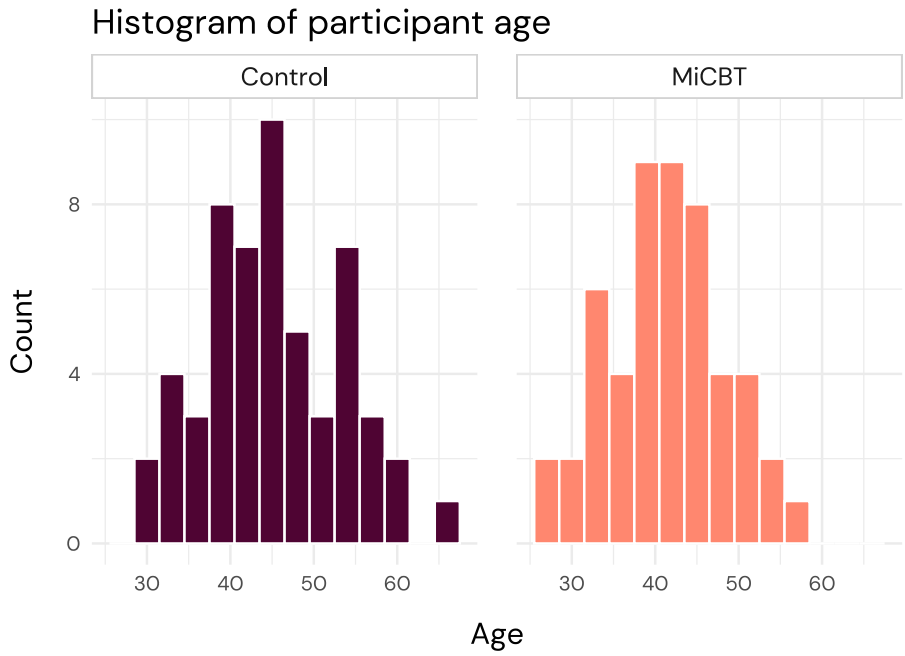
- dynamic color/fill (based on a data variable) needs to be defined within the `aes()` function (aesthetics).

```
ggplot(df,
       aes(x = age, fill = Group)) +
  geom_histogram(color = "black") +
  labs(title = "Histogram of participant age",
       x = "Age",
       y = "Count") +
  theme_ki() +
  scale_y_continuous(breaks = c(0,4,8,12)) +
  scale_color_manual(values = ki_color_palette,
                     aesthetics = c("fill","color"))
```



Or we can use `facet_wrap()` to make parallel plots.

```
ggplot(df,
       aes(x = age, fill = Group)) +
  geom_histogram(color = "white",
                 binwidth = 3) +
  labs(title = "Histogram of participant age",
       x = "Age",
       y = "Count") +
  theme_ki() +
  scale_y_continuous(breaks = c(0,4,8,12)) +
  scale_color_manual(values = ki_color_palette,
                     aesthetics = c("fill","color"),
                     guide = "none") +
  facet_wrap(~Group)
```



Variable names

```
names(df)
```

```
 [1] "id"                "Group"             "K10_Score_GP"
 [4] "GPK10_coded"       "TK10_t0"           "K10preCat"
 [7] "Practice"          "Sessions"          "Age_bands"
[10] "Age_split"         "qualifications"    "Medications_split"
[13] "Meditation"        "Temotreg_t0"       "Tequanimity_t0"
[16] "Tsocialskills_t0"  "Tdistresstol_t0"   "Ttakerespons_t0"
[19] "Tinterpersonal_t0" "Tnoticing_t0"      "Tnotdisract_t0"
[22] "Tnotworry_t0"      "Tattenreg_t0"      "Temotaware_t0"
[25] "Tbodylisten_t0"    "Ttrusting_t0"      "Tselfreg_t0"
[28] "TNAS_t0"           "TNAS_MEAN_t0"      "EQ_t0"
[31] "TFs_t0"            "TSWLS_t0"          "TDASS_t0"
[34] "TANX_t0"           "TSTRESS_t0"        "TDEP_t0"
[37] "TMSES_t0"          "Temotreg_t1"       "Tequanimity_t1"
[40] "Tsocialskills_t1"  "Tdistresstol_t1"   "Ttakerespons_t1"
[43] "Tinterpersonal_t1" "TMSES_t1"          "Tnoticing_t1"
[46] "Tnotdisract_t1"    "Tnotworry_t1"      "Tattenreg_t1"
[49] "Temotaware_t1"     "Tbodylisten_t1"    "Ttrusting_t1"
[52] "Tselfreg_t1"       "TNAS_t1"           "TNAS_MEAN_t1"
[55] "EQ_t1"             "TFs_t1"            "TSWLS_t1"
[58] "TK10_t1"           "TDASS_t1"          "TANX_t1"
[61] "TSTRESS_t1"        "TDEP_t1"           "A8week_program"
[64] "Temotreg_t2"       "Tequanimity_t2"    "Tsocialskills_t2"
[67] "Tdistresstol_t2"   "Ttakerespons_t2"   "Tinterpersonal_t2"
```

```
 [70] "TMSES_t2"          "Tnoticing_t2"      "Tnotdisract_t2"
 [73] "Tnotworry_t2"      "Tattenreg_t2"      "Temotaware_t2"
 [76] "Tbodylisten_t2"    "Ttrusting_t2"      "Tselfreg_t2"
 [79] "TNAS_t2"           "TNAS_MEAN_t2"      "EQ_t2"
 [82] "TFs_t2"            "TSWLS_t2"          "TK10_t2"
 [85] "TDASS_t2"          "TANX_t2"           "TSTRESS_t2"
 [88] "TDEP_t2"           "Completed_program" "medication_change"
 [91] "Temotreg_t3"       "Tequanimity_t3"    "Tsocialskills_t3"
 [94] "Tdistresstol_t3"   "Ttakerespons_t3"   "Tinterpersonal_t3"
 [97] "TMSES_t3"          "Tnoticing_t3"      "Tnotdisract_t3"
[100] "Tnotworry_t3"      "Tattenreg_t3"      "Temotaware_t3"
[103] "Tbodylisten_t3"    "Ttrusting_t3"      "Tselfreg_t3"
[106] "TNAS_t3"           "TNAS_MEAN_t3"      "EQ_t3"
[109] "TFs_t3"            "TSWLS_t3"          "TK10_t3"
[112] "TDASS_t3"          "TANX_t3"           "TSTRESS_t3"
[115] "TDEP_t3"           "TMAIA_t0"          "TMAIA_t1"
[118] "TMAIA_t2"          "TMAIA_t3"          "MSES_Et0"
[121] "MSES_Et1"          "MSES_Et2"          "MSES_Et3"
[124] "MSES_It0"          "MSES_It1"          "MSES_It2"
[127] "MSES_It3"          "age"               "sex"
```

Why is there a "T" pretty much everywhere? What happens if we remove it?

```
df %>%
  rename_all(~ str_replace(.x, "^T","")) %>%
  names()
```

```
 [1] "id"                "Group"             "K10_Score_GP"
 [4] "GPK10_coded"       "K10_t0"            "K10preCat"
 [7] "Practice"          "Sessions"          "Age_bands"
[10] "Age_split"         "qualifications"    "Medications_split"
[13] "Meditation"        "emotreg_t0"        "equanimity_t0"
[16] "socialskills_t0"   "distresstol_t0"    "takerespons_t0"
[19] "interpersonal_t0"  "noticing_t0"       "notdisract_t0"
[22] "notworry_t0"       "attenreg_t0"       "emotaware_t0"
[25] "bodylisten_t0"     "trusting_t0"       "selfreg_t0"
[28] "NAS_t0"            "NAS_MEAN_t0"       "EQ_t0"
[31] "Fs_t0"             "SWLS_t0"           "DASS_t0"
[34] "ANX_t0"            "STRESS_t0"         "DEP_t0"
[37] "MSES_t0"           "emotreg_t1"        "equanimity_t1"
[40] "socialskills_t1"   "distresstol_t1"    "takerespons_t1"
[43] "interpersonal_t1"  "MSES_t1"           "noticing_t1"
[46] "notdisract_t1"     "notworry_t1"       "attenreg_t1"
[49] "emotaware_t1"      "bodylisten_t1"     "trusting_t1"
[52] "selfreg_t1"        "NAS_t1"            "NAS_MEAN_t1"
[55] "EQ_t1"             "Fs_t1"             "SWLS_t1"
[58] "K10_t1"            "DASS_t1"           "ANX_t1"
[61] "STRESS_t1"         "DEP_t1"            "A8week_program"
[64] "emotreg_t2"        "equanimity_t2"     "socialskills_t2"
[67] "distresstol_t2"    "takerespons_t2"    "interpersonal_t2"
```

```
 [70] "MSES_t2"           "noticing_t2"       "notdisract_t2"
 [73] "notworry_t2"       "attenreg_t2"       "emotaware_t2"
 [76] "bodylisten_t2"     "trusting_t2"       "selfreg_t2"
 [79] "NAS_t2"            "NAS_MEAN_t2"       "EQ_t2"
 [82] "Fs_t2"             "SWLS_t2"           "K10_t2"
 [85] "DASS_t2"           "ANX_t2"            "STRESS_t2"
 [88] "DEP_t2"            "Completed_program" "medication_change"
 [91] "emotreg_t3"        "equanimity_t3"     "socialskills_t3"
 [94] "distresstol_t3"    "takerespons_t3"    "interpersonal_t3"
 [97] "MSES_t3"           "noticing_t3"       "notdisract_t3"
[100] "notworry_t3"       "attenreg_t3"       "emotaware_t3"
[103] "bodylisten_t3"     "trusting_t3"       "selfreg_t3"
[106] "NAS_t3"            "NAS_MEAN_t3"       "EQ_t3"
[109] "Fs_t3"             "SWLS_t3"           "K10_t3"
[112] "DASS_t3"           "ANX_t3"            "STRESS_t3"
[115] "DEP_t3"            "MAIA_t0"           "MAIA_t1"
[118] "MAIA_t2"           "MAIA_t3"           "MSES_Et0"
[121] "MSES_Et1"          "MSES_Et2"          "MSES_Et3"
[124] "MSES_It0"          "MSES_It1"          "MSES_It2"
[127] "MSES_It3"          "age"               "sex"
```

Wide to long format

Almost everything in R likes long format. What is the naming scheme?

Let's look at the variables ending with "t".

```
df %>%
  select(ends_with("t0")) %>%
  names()
```

```
 [1] "TK10_t0"           "Temotreg_t0"       "Tequanimity_t0"
 [4] "Tsocialskills_t0"  "Tdistresstol_t0"   "Ttakerespons_t0"
 [7] "Tinterpersonal_t0" "Tnoticing_t0"      "Tnotdisract_t0"
[10] "Tnotworry_t0"      "Tattenreg_t0"      "Temotaware_t0"
[13] "Tbodylisten_t0"    "Ttrusting_t0"      "Tselfreg_t0"
[16] "TNAS_t0"           "TNAS_MEAN_t0"      "EQ_t0"
[19] "TFs_t0"            "TSWLS_t0"          "TDASS_t0"
[22] "TANX_t0"           "TSTRESS_t0"        "TDEP_t0"
[25] "TMSES_t0"          "TMAIA_t0"          "MSES_Et0"
[28] "MSES_It0"
```

```
df %>%
  select(ends_with("t1")) %>%
  names()
```

```
 [1] "Temotreg_t1"       "Tequanimity_t1"    "Tsocialskills_t1"
 [4] "Tdistresstol_t1"   "Ttakerespons_t1"   "Tinterpersonal_t1"
 [7] "TMSES_t1"          "Tnoticing_t1"      "Tnotdisract_t1"
[10] "Tnotworry_t1"      "Tattenreg_t1"      "Temotaware_t1"
```

```
[13] "Tbodylisten_t1"    "Ttrusting_t1"      "Tselfreg_t1"
[16] "TNAS_t1"           "TNAS_MEAN_t1"      "EQ_t1"
[19] "TFs_t1"            "TSWLS_t1"          "TK10_t1"
[22] "TDASS_t1"          "TANX_t1"           "TSTRESS_t1"
[25] "TDEP_t1"           "TMAIA_t1"          "MSES_Et1"
[28] "MSES_It1"
```

Seems like some variables don't stick to the `_t[0-3]` scheme…

We can demonstrate the option-click cursor in the example below.

```
df.wide <- df %>%
  rename(MSESe_t0 = MSES_Et0,
         MSESi_t0 = MSES_It0,
         MSESe_t1 = MSES_Et1,
         MSESi_t1 = MSES_It1,
         MSESe_t2 = MSES_Et2,
         MSESi_t2 = MSES_It2,
         MSESe_t3 = MSES_Et3,
         MSESi_t3 = MSES_It3)
```

Let's try to pivot this dataframe to long format.

```
df.long <- df.wide %>%
  pivot_longer(ends_with(c("t0","t1","t2","t3")),
               names_to = c("measure", "time"),
               names_sep = "_t")
```

And look at the df.

```
glimpse(df.long)
```

```
Rows: 11,872
Columns: 20
$ id                <dbl> 830, 830, 830, 830, 830, 830, 830, 830, 830, 830, 83~
$ Group             <chr> "Control", "Control", "Control", "Control", "Control~
$ K10_Score_GP      <dbl> 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, ~
$ GPK10_coded       <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
$ K10preCat         <chr> "30+", "30+", "30+", "30+", "30+", "30+", "30+", "30~
$ Practice          <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
$ Sessions          <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
$ Age_bands         <chr> "35-39", "35-39", "35-39", "35-39", "35-39", "35-39"~
$ Age_split         <chr> "35-49", "35-49", "35-49", "35-49", "35-49", "35-49"~
$ qualifications    <chr> "Yes has further qualifications", "Yes has further q~
$ Medications_split <chr> "Yes", "Yes", "Yes", "Yes", "Yes", "Yes", "Yes", "Ye~
$ Meditation        <chr> "Yes", "Yes", "Yes", "Yes", "Yes", "Yes", "Yes", "Ye~
$ A8week_program    <chr> "No", "No", "No", "No", "No", "No", "No", "No", "No"~
$ Completed_program <chr> "No", "No", "No", "No", "No", "No", "No", "No", "No"~
$ medication_change <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, ~
$ age               <dbl> 50, 50, 50, 50, 50, 50, 50, 50, 50, 50, 50, 50, 50, ~
```

```
$ sex               <int> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2~
$ measure           <chr> "TK10", "Temotreg", "Tequanimity", "Tsocialskills", ~
$ time              <chr> "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0~
$ value             <dbl> 30.000000, 8.000000, 6.000000, 10.000000, 9.000000, ~
```

Descriptives

Correlation & visualization

rework these for new dataset, first without "styling"

Correlation matrix

without and with grouping

```
df %>%
  select(starts_with("WHO"),GDS,WHOQoL_BREF,NMM,Gender) %>%
  ggpairs(aes(color = Gender),alpha = 0.8) +
  scale_color_manual(values = RISEpalette1,
                     aesthetics = c("color", "fill"))
```

easystats `plot(cor_test())`

```
plot(cor_test(df, "NMM", "WHOQoL_BREF")) +
  theme_minimal(base_size = 15,
                base_family = "Lato") +
  theme_rise(fontfamily = "Lato",
             axissize = 15) +
  geom_point(data = df,
             aes(NMM, WHOQoL_BREF),
             size = 2) +
    ylab("WHOQoL-BREF") +
    xlab("NMM")
```