micro RNA

Program 07

Francois Collin

2022-06-08

Table of contents

Preamble

Development version:

- only the program is displayed within these pages.
- no data is attached to the repository or displayed within the pages.
- no output is displayed within the pages.

Outputs will be included and made available within the program if the associated manuscript is accepted for publication in peer-review journal.

1 Missingness in adlb and advs

```
params <- yaml::read_yaml("_prog.yml")
devtools::load_all("src/pkg/dbs.data")
```

```
i Loading dbs.data
```

```
devtools::load_all("src/pkg/latarnia.utils")
```

```
i Loading latarnia.utils
```

```
Loading required package: grid
```

```
Loading required package: shiny
```

```
knitr::opts_chunk$set(results = params$knitr$results)
```

```
adsl <- dbs.data::adsl
advs <- dbs.data::advs
adlb <- dbs.data::adlb
```

```
library(ggplot2)
gg <- rbind(adlb, advs) |>
  ggplot(aes(subjid, paramcd, fill = dtype)) +
  scale_fill_manual(values = c("gray", "orange")) +
  geom_tile(color = "gray50") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```
gg
```

2 Demographics and Baseline Anthropometrics

2.1 Demographics and Baseline Anthropometrics

Target:

- ☒ Table: Demographics and baseline anthropometrics are tested via an Anova.
- ☒ Supp. Table: Post-hoc estimations / tests by diabetes groups.
- ☒ Supp. Table: extension of the anova to additional ADVS/ADLB parameters.

```
params <- yaml::read_yaml("_prog.yml")
devtools::load_all("src/pkg/dbs.data")
```

```
i Loading dbs.data
```

```
devtools::load_all("src/pkg/latarnia.utils")
```

```
i Loading latarnia.utils
```

```
Loading required package: grid
```

```
Loading required package: shiny
```

```
knitr::opts_chunk$set(results = params$knitr$results)
```

```
adsl <- dbs.data::adsl
advs <- dbs.data::advs
adlb <- dbs.data::adlb
```

```r
ads <- adlb |>
  rbind(advs) |>
  subset(dtype == "" & avisit == "Baseline") |>
  subset(select = c(subjid, paramcd, avisit, aval, dtype)) |>
  rbind(
    adsl |>
      subset(select = -c(diab, diabcd)) |>
      tidyr::pivot_longer(
        cols = c("age", "trainn"), names_to = "paramcd", values_to = "aval"
      ) |>
      within(dtype <- "") |>
      within(avisit <- "Baseline")
  ) |>
  (\(x) merge(x = adsl[c("subjid", "diabcd")], y = x, by = "subjid"))() |>
  (\(df, fct = "diabcd") {
    df[paste0(fct, "_n")] <- factor_n(df, fct, id = "subjid", sep = " ")
    df
  })() |>
  within(
    paramcd <- factor(
      paramcd,
      levels = c(
        "age", "trainn", "GLU0", "GLU30", "GLU60", "GLU120",
        "INSULIN0", "INSULIN30", "INSULIN60", "INSULIN120",
        "HBA1C", "HOMAB", "HOMAIR", "MATSUDA",
        "TRIG", "CHOL", "LDL", "HDL", "VO2MAXE", "WEIGHT", "BMI",
        "FATMASS", "BODYFATP", "LEANMASS", "VAT",
        "DCAL", "FATACFR", "DCARBT", "DFATT", "DPROT", "SMMASS",
        "VO2MAXLBM", "VO2MAXML"
      )
    )
  ) |>
  (\(df) df[order(df$paramcd), ])()
```

```r
format_pval <- function(x) {
  p <- round(x, 5)
  ifelse(
    test = p < 0.0001,
    yes = "<0.0001",
    no = ifelse(
      test = p < 0.001,
```

```
      yes = format(round(p, 4), nsmall = 4),
      no = ifelse(
        test = p < 0.01,
        yes = format(round(p, 3), nsmall = 3),
        no = format(round(p, 2), nsmall = 2)
      )
    )
  )
}

lm_by_paramcd <- function(x,
                          dep_var = "aval",
                          indep_var = "diabcd_n",
                          covariate = NULL) {

  formula <- paste(
    dep_var, "~",
    if (!is.null(covariate)) paste(covariate, "+"),
    indep_var
  )
  formula <- as.formula(formula)

  lapply(
    x,
    \(x) list(data = x, lm = lm(formula, data = x))
  )
}

make_specs <- function(var) as.formula(paste("~", var))
lsm_by_param <- function(x, indep_var = "diabcd_n") {
  lapply(
    x,
    \(x) {
      mod_em <- emmeans::emmeans(x$lm, specs = make_specs(indep_var))
      y <- multcomp::cld(mod_em, Letters = letters)
      y <- as.data.frame(y)
      cbind(
        paramcd = unique(x$data$paramcd),
        y,
        diabcd_f = car::Anova(x$lm)[indep_var, "Pr(>F)"]
      )
```

```
    }
  )
}

lsm_pairs_by_param <- function(x, indep_var = "diabcd_n")
  lapply(
    x,
    \(x) {
      mod_em <- emmeans::emmeans(
        x$lm, specs = indep_var, contr = "revpairwise"
      )
      y <- merge(
        as.data.frame(mod_em$contrast)[c("contrast", "p.value")],
        confint(mod_em)$contrasts
      )
      cbind(paramcd = unique(x$data$paramcd), y)
    }
  )
```

2.2 Tab 08 01 - Demographics and Baseline Anthropometrics by Diabetes Group

```
tab_08_01_raw <- ads |>
  subset(
    paramcd %in% c(
      "age", "trainn", "GLU0", "GLU30", "GLU60", "GLU120",
      "INSULIN0", "INSULIN30", "INSULIN60", "INSULIN120",
      "HBA1C", "HOMAB", "HOMAIR", "MATSUDA",
      "TRIG", "CHOL", "LDL", "HDL", "VO2MAXE", "WEIGHT", "BMI",
      "FATMASS", "BODYFATP", "LEANMASS", "VAT"
    )
  ) |>
  (\(x) split(x, f = x$paramcd, drop = TRUE))() |>
  lm_by_paramcd() |>
  lsm_by_param() |>
  (\(x) Reduce(rbind, x))()

library(tidyr)
```

```
Attaching package: 'tidyr'

The following object is masked from 'package:testthat':

    matches
```

```
tab_08_01 <- tab_08_01_raw |>
  (\(df) df[order(df$diabcd_n), ])() |>
  within({
    val <- paste0(
      signif(emmean, 3),
      " (", signif(lower.CL, 3), ", ", signif(upper.CL, 3), ")"
    )
    pval <- format_pval(diabcd_f)
  }) |>
  pivot_wider(
    id_cols = c("paramcd", "pval", "diabcd_f"),
    values_from = "val",
    names_from = "diabcd_n"
  )
tab_08_01
```

```
library(flextable)
```

```
Attaching package: 'flextable'

The following objects are masked from 'package:latarnia.utils':

    add_footer, add_header
```

```
tab_08_01_ft <- tab_08_01 |>
  subset(select = -diabcd_f) |>
  flextable() |>
  autofit() |>
  add_header_lines(wrap_long_lines(
    "Analysis Set: Full Analysis Set - Observed Cases at baseline"
  )) |>
```

```
  set_caption(
    caption = wrap_long_lines(
      "Tab 08 01 - Analysis of Variance / Least Means Square estimations
      (95% Confidence Interval) of Demographics Parameters and Baseline
      Anthropometrics by Diabetes Group"
  )
  ) |>
  footnote(
    part = "header",
    i = 2, j = 2,
    value = as_paragraph(
      "Note: pval, p value of diabetes group effect test by F test."
    ),
    ref_symbols = "a"
  ) |>
  footnote(
    value = as_paragraph(
      "Source: ADSL and ADVS/ADLB observed cases at baseline."
    ),
    ref_symbols = ""
  ) |>
  theme_booktabs()
tab_08_01_ft
```

```
Warning: Warning: fonts used in `flextable` are ignored because the `pdflatex`
engine is used and not `xelatex` or `lualatex`. You can avoid this warning
by using the `set_flextable_defaults(fonts_ignore=TRUE)` command or use a
compatible engine by defining `latex_engine: xelatex` in the YAML header of the
R Markdown document.
```

```
bnm <- "tab_08_01"
dir_tab <- params$paths$tab
dir_dta <- params$paths$dta

file.path(dir_tab, paste0(bnm, "_ft.RData")) %T>%
  message("[output] Table saved as ", .) %>%
  save(tab_08_01_ft, file = .)
```

```
[output] Table saved as ../tlg/tables/tab_08_01_ft.RData
```

```
  file.path(dir_dta, paste0(bnm, ".RData")) %T>%
    message("[output] Table saved as ", .) %>%
    save(tab_08_01, file = .)
```

```
[output] Table saved as ../data/tab_08_01.RData
```

```
  file.path(dir_tab, paste(bnm, sep = ".", "docx")) %T>%
    message("[output] Table saved as ", .) %>%
    save_as_docx(tab_08_01_ft, path = .)
```

```
[output] Table saved as ../tlg/tables/tab_08_01.docx
```

```
  file.path(dir_tab, paste(bnm, sep = ".", "html")) %T>%
    message("[output] Table saved as ", .) %>%
    save_as_html(tab_08_01_ft, path = .)
```

```
[output] Table saved as ../tlg/tables/tab_08_01.html
```

```
  file.path(dir_dta, paste(bnm, sep = ".", "csv")) %T>%
    message("[output] Table saved as ", .) %>%
    write.csv(tab_08_01, file = ., row.names = FALSE)
```

```
[output] Table saved as ../data/tab_08_01.csv
```

2.3 Tab 08 02 - Post-hoc: Demographics and Baseline Anthropometrics by Diabetes Group

```
tab_08_02_raw <- ads |>
  subset(
    paramcd %in% c(
      "age", "trainn", "GLU0", "GLU30", "GLU60", "GLU120",
      "INSULIN0", "INSULIN30", "INSULIN60", "INSULIN120",
      "HBA1C", "HOMAB", "HOMAIR", "MATSUDA",
```

```
      "TRIG", "CHOL", "LDL", "HDL", "VO2MAXE", "WEIGHT", "BMI",
      "FATMASS", "BODYFATP", "LEANMASS", "VAT"
    )
  ) |>
  (\(x) split(x, f = x$paramcd, drop = TRUE))() |>
  lm_by_paramcd(indep_var = "diabcd") |>
  lsm_pairs_by_param(indep_var = "diabcd")  |>
  (\(x) Reduce(rbind, x))()

library(tidyr)
tab_08_02 <- tab_08_02_raw |>
  subset(select = c(
    paramcd, contrast, estimate, SE, df, p.value, lower.CL, upper.CL
  ))

library(flextable)
tab_08_02_ft <- tab_08_02 |>
  (\(x) split(x, f = x$paramcd))() |>
  lapply(
    \(x) {
      x$p.value <- format(round(x$p.value, 5))
      x$estimate <- format(signif(x$estimate, 5))
      x$SE <- format(signif(x$SE, 6))
      x$lower.CL <- format(signif(x$lower.CL, 5))
      x$upper.CL <- format(signif(x$upper.CL, 5))
      x
    }) |>
  (\(x) Reduce(rbind, x))()|>
  flextable() |>
  fontsize(size = 9, part = "all") |>
  autofit() |>
    add_header_lines(
    "Analysis Set: Full Analysis Set - Observed Cases at baseline"
  ) |>
  set_caption(
    caption = wrap_long_lines(
      "Tab 08 02 - Post-hoc tests for the Analysis of Variance of
      Demographics and Baseline Anthropometrics by Diabetes Group"
  )
  ) |>
  footnote(
```

```
    value = as_paragraph(wrap_long_lines(
      "CL, 95% Confidence Limit; SE, Standard Error."
    )),
    ref_symbols = ""
  ) |>
  footnote(
    value = as_paragraph(wrap_long_lines(
      "Note: P value adjustment by Tukey's method for comparing a family of
      3 estimates."
    )),
    ref_symbols = ""
  )|>
    footnote(
    value = as_paragraph(wrap_long_lines(
      "Source: ADSL and ADVS/ADLB observed cases at
      baseline."
    )),
    ref_symbols = ""
  ) |>
  theme_booktabs()

tab_08_02_ft
```

```
Warning: Warning: fonts used in `flextable` are ignored because the `pdflatex`
engine is used and not `xelatex` or `lualatex`. You can avoid this warning
by using the `set_flextable_defaults(fonts_ignore=TRUE)` command or use a
compatible engine by defining `latex_engine: xelatex` in the YAML header of the
R Markdown document.
```

```
bnm <- "tab_08_02"
dir_tab <- params$paths$tab
dir_dta <- params$paths$dta

file.path(dir_tab, paste0(bnm, "_ft.RData")) %T>%
  message("[output] Table saved as ", .) %>%
  save(tab_08_02_ft, file = .)
```

```
[output] Table saved as ../tlg/tables/tab_08_02_ft.RData
```

```
  file.path(dir_dta, paste0(bnm, ".RData")) %T>%
    message("[output] Table saved as ", .) %>%
    save(tab_08_02, file = .)
```

```
[output] Table saved as ../data/tab_08_02.RData
```

```
  file.path(dir_tab, paste(bnm, sep = ".", "docx")) %T>%
    message("[output] Table saved as ", .) %>%
    save_as_docx(tab_08_02_ft, path = .)
```

```
[output] Table saved as ../tlg/tables/tab_08_02.docx
```

```
  file.path(dir_tab, paste(bnm, sep = ".", "html")) %T>%
    message("[output] Table saved as ", .) %>%
    save_as_html(tab_08_02_ft, path = .)
```

```
[output] Table saved as ../tlg/tables/tab_08_02.html
```

```
  file.path(dir_dta, paste(bnm, sep = ".", "csv")) %T>%
    message("[output] Table saved as ", .) %>%
    write.csv(tab_08_02, file = ., row.names = FALSE)
```

```
[output] Table saved as ../data/tab_08_02.csv
```

2.4 Tab 08 03 - Demographics and Baseline Anthropometrics by Diabetes Group (additional parameters)

```
tab_08_03_raw <- ads |>
  subset(
    !paramcd %in% c(
      "age", "trainn", "GLU0", "GLU30", "GLU60", "GLU120",
      "INSULIN0", "INSULIN30", "INSULIN60", "INSULIN120",
      "HBA1C", "HOMAB", "HOMAIR", "MATSUDA",
```

```
      "TRIG", "CHOL", "LDL", "HDL", "VO2MAXE", "WEIGHT", "BMI",
      "FATMASS", "BODYFATP", "LEANMASS", "VAT"
    )
  ) |>
  (\(x) split(x, f = x$paramcd, drop = TRUE))() |>
  lm_by_paramcd() |>
  lsm_by_param() |>
  (\(x) Reduce(rbind, x))()

library(tidyr)
tab_08_03 <- tab_08_03_raw |>
  (\(df) df[order(df$diabcd_n), ])() |>
  within({
    val <- paste0(
      signif(emmean, 3),
      " (", signif(lower.CL, 3), ", ", signif(upper.CL, 3), ")"
    )
    pval <- format_pval(diabcd_f)
  }) |>
  pivot_wider(
    id_cols = c("paramcd", "pval", "diabcd_f"),
    values_from = "val",
    names_from = "diabcd_n"
  )
tab_08_03
```

```
library(flextable)
tab_08_03_ft <- tab_08_03 |>
  subset(select = -diabcd_f) |>
  flextable() |>
  autofit() |>
  add_header_lines(wrap_long_lines(
    "Analysis Set: Full Analysis Set - Observed Cases at baseline"
  )) |>
  set_caption(
    caption = wrap_long_lines(
      "Tab 08 03 - Analysis of Variance / Least Means Square estimations
      (95% Confidence Interval) of Demographics Parameters and Baseline
      Anthropometrics by Diabetes Group for Supplementary Parameters"
  )
  ) |>
```

```
  footnote(
    part = "header",
    i = 2, j = 2,
    value = as_paragraph(
      "Note: pval, p value of diabetes group effect test by F test."
    ),
    ref_symbols = "a"
  ) |>
  footnote(
    value = as_paragraph(
      "Source: ADSL and ADVS/ADLB observed cases at baseline."
    ),
    ref_symbols = ""
  ) |>
  theme_booktabs()
tab_08_03_ft
```

```
Warning: Warning: fonts used in `flextable` are ignored because the `pdflatex`
engine is used and not `xelatex` or `lualatex`. You can avoid this warning
by using the `set_flextable_defaults(fonts_ignore=TRUE)` command or use a
compatible engine by defining `latex_engine: xelatex` in the YAML header of the
R Markdown document.
```

```
bnm <- "tab_08_03"
dir_tab <- params$paths$tab
dir_dta <- params$paths$dta

file.path(dir_tab, paste0(bnm, "_ft.RData")) %T>%
  message("[output] Table saved as ", .) %>%
  save(tab_08_03_ft, file = .)
```

```
[output] Table saved as ../tlg/tables/tab_08_03_ft.RData
```

```
file.path(dir_dta, paste0(bnm, ".RData")) %T>%
  message("[output] Table saved as ", .) %>%
  save(tab_08_03, file = .)
```

```
[output] Table saved as ../data/tab_08_03.RData
```

```
  file.path(dir_tab, paste(bnm, sep = ".", "docx")) %T>%
    message("[output] Table saved as ", .) %>%
    save_as_docx(tab_08_03_ft, path = .)
```

```
[output] Table saved as ../tlg/tables/tab_08_03.docx
```

```
  file.path(dir_tab, paste(bnm, sep = ".", "html")) %T>%
    message("[output] Table saved as ", .) %>%
    save_as_html(tab_08_03_ft, path = .)
```

```
[output] Table saved as ../tlg/tables/tab_08_03.html
```

```
  file.path(dir_dta, paste(bnm, sep = ".", "csv")) %T>%
    message("[output] Table saved as ", .) %>%
    write.csv(tab_08_03, file = ., row.names = FALSE)
```

```
[output] Table saved as ../data/tab_08_03.csv
```

2.5 Tab 08 04 - Post-hoc: Demographics and Baseline Anthropometrics by Diabetes Group (additional parameters)

```
tab_08_04_raw <- ads |>
  subset(
    ! paramcd %in% c(
      "age", "trainn", "GLU0", "GLU30", "GLU60", "GLU120",
      "INSULIN0", "INSULIN30", "INSULIN60", "INSULIN120",
      "HBA1C", "HOMAB", "HOMAIR", "MATSUDA",
      "TRIG", "CHOL", "LDL", "HDL", "VO2MAXE", "WEIGHT", "BMI",
      "FATMASS", "BODYFATP", "LEANMASS", "VAT"
    )
  ) |>
  (\(x) split(x, f = x$paramcd, drop = TRUE))() |>
  lm_by_paramcd(indep_var = "diabcd") |>
  lsm_pairs_by_param(indep_var = "diabcd")  |>
  (\(x) Reduce(rbind, x))()
```

```r
library(tidyr)
tab_08_04 <- tab_08_04_raw |>
  subset(select = c(
    paramcd, contrast, estimate, SE, df, p.value, lower.CL, upper.CL
  ))

library(flextable)
tab_08_04_ft <- tab_08_04 |>
  (\(x) split(x, f = x$paramcd))() |>
  lapply(
    \(x) {
      x$p.value <- format(round(x$p.value, 5))
      x$estimate <- format(signif(x$estimate, 5))
      x$SE <- format(signif(x$SE, 6))
      x$lower.CL <- format(signif(x$lower.CL, 5))
      x$upper.CL <- format(signif(x$upper.CL, 5))
      x
    }) |>
  (\(x) Reduce(rbind, x))()|>
  flextable() |>
  fontsize(size = 9, part = "all") |>
  autofit() |>
    add_header_lines(
    "Analysis Set: Full Analysis Set - Observed Cases at baseline"
  ) |>
  set_caption(
    caption = wrap_long_lines(
      "Tab 08 04 - Post-hoc tests for the Analysis of Variance of
      Demographics and Baseline Anthropometrics by Diabetes Group"
  )
  ) |>
  footnote(
    value = as_paragraph(wrap_long_lines(
      "CL, 95% Confidence Limit; SE, Standard Error."
    )),
    ref_symbols = ""
  ) |>
  footnote(
    value = as_paragraph(wrap_long_lines(
      "Note: P value adjustment by Tukey's method for comparing a family of
      3 estimates."
```

```
    )),
    ref_symbols = ""
  )|>
    footnote(
    value = as_paragraph(wrap_long_lines(
      "Source: ADSL and ADVS/ADLB observed cases at
      baseline."
    )),
    ref_symbols = ""
  ) |>
  theme_booktabs()

tab_08_04_ft
```

```
Warning: Warning: fonts used in `flextable` are ignored because the `pdflatex`
engine is used and not `xelatex` or `lualatex`. You can avoid this warning
by using the `set_flextable_defaults(fonts_ignore=TRUE)` command or use a
compatible engine by defining `latex_engine: xelatex` in the YAML header of the
R Markdown document.
```

```
bnm <- "tab_08_04"
dir_tab <- params$paths$tab
dir_dta <- params$paths$dta

file.path(dir_tab, paste0(bnm, "_ft.RData")) %T>%
  message("[output] Table saved as ", .) %>%
  save(tab_08_04_ft, file = .)
```

```
[output] Table saved as ../tlg/tables/tab_08_04_ft.RData
```

```
file.path(dir_dta, paste0(bnm, ".RData")) %T>%
  message("[output] Table saved as ", .) %>%
  save(tab_08_04, file = .)
```

```
[output] Table saved as ../data/tab_08_04.RData
```

```
  file.path(dir_tab, paste(bnm, sep = ".", "docx")) %T>%
    message("[output] Table saved as ", .) %>%
    save_as_docx(tab_08_04_ft, path = .)
```

```
[output] Table saved as ../tlg/tables/tab_08_04.docx
```

```
  file.path(dir_tab, paste(bnm, sep = ".", "html")) %T>%
    message("[output] Table saved as ", .) %>%
    save_as_html(tab_08_04_ft, path = .)
```

```
[output] Table saved as ../tlg/tables/tab_08_04.html
```

```
  file.path(dir_dta, paste(bnm, sep = ".", "csv")) %T>%
    message("[output] Table saved as ", .) %>%
    write.csv(tab_08_04, file = ., row.names = FALSE)
```

```
[output] Table saved as ../data/tab_08_04.csv
```

3 Anthropometrics Change From Baseline

3.1 Tab 09 01 - Ancova - Anthropometrics Changes from Baseline by Diabetes Group

Questions: for the analysis of a post-treatment values, should we analyse the change from baseline or percentage change from baseline? Should we adjust for the baseline?

Back to 2003, in the context of randomized clinical trial, the European Medicines Agency (COMMITTEE FOR PROPRIETARY MEDICINAL PRODUCTS 2003), when the endpoint is studied as a change from baseline, the adjustment for baseline improves the accuracy in comparison to non-baseline adjustment; estimates becomes also equivalent to the standard linear model, the choice of change from baseline analysis or raw value is then only a question of interpretability. They renewed the recommendation in 2015 (Committee for Medicinal Products for Human Use (CHMP) 2015).

From the academic side, the topic was repeatedly studied:

- Van Breukelen (2006): > *"In randomized studies both methods [Anova, no BL adjustment vs Ancova] > are unbiased, but ANCOVA has more power"*
- Liu et al. (2009) also highlighted the benefits of adjustment for baseline as a covariate.
- this was later confirmed by Zhang et al. (2014).
- More recently O'Connell et al. (2017) also defended the superiority of the Ancova-change: $y_i = \beta_0 + \beta_1 X_i + \beta_2 Y_{0i,=BL} + \varepsilon_i$

  "Consistent with existing literature, our results demonstrate that each method leads to unbiased treatment effect estimates, and based on precision of estimates, 95% coverage probability, and power, ANCOVA modeling of either change scores or post-treatment score as the outcome, prove to be the most effective."

Most of the authors above are specifically working on randomized trial, Vickers (2001) also brought some light on the topic, and highlighted in addition that: working with percentage change is generally a bad idea. The extended to a theoretical works also indicated that the percentage change from baseline *"will also fail to protect from bias in the case of baseline imbalance and will lead to an excess of trials with non-normally distributed outcome data"*.

```
params <- yaml::read_yaml("_prog.yml")
devtools::load_all("src/pkg/dbs.data")
```

```
i Loading dbs.data
```

```
devtools::load_all("src/pkg/latarnia.utils")
```

```
i Loading latarnia.utils
```

```
Loading required package: grid
```

```
Loading required package: shiny
```

```
knitr::opts_chunk$set(results = params$knitr$results)
```

```
adsl <- dbs.data::adsl
advs <- dbs.data::advs
adlb <- dbs.data::adlb
```

```
ads <- adlb |>
  rbind(advs) |>
  subset(basetype == "" & avisit != "Baseline") |>
  subset(select = c(subjid, paramcd, avisit, base, chg)) |>
  (\(x) merge(x = adsl[c("subjid", "diabcd")], y = x, by = "subjid"))() |>
  (\(df, fct = "diabcd") {
    df[paste0(fct, "_n")] <- factor_n(df, fct, id = "subjid", sep = " ")
    df
  })()

head(ads)
```

```
format_pval <- function(x) {
  p <- round(x, 5)
  ifelse(
    test = p < 0.0001,
    yes = "<0.0001",
```

```
    no = ifelse(
      test = p < 0.001,
      yes = format(round(p, 4), nsmall = 4),
      no = ifelse(
        test = p < 0.01,
        yes = format(round(p, 3), nsmall = 3),
        no = format(round(p, 2), nsmall = 2)
      )
    )
  )
}

lm_by_paramcd <- function(x,
                          dep_var = "aval",
                          indep_var = "diabcd_n",
                          covariate = NULL) {

  formula <- paste(
    dep_var, "~",
    if (!is.null(covariate)) paste(covariate, "+"),
    indep_var
  )
  formula <- as.formula(formula)

  lapply(
    x,
    \(x) list(data = x, lm = lm(formula, data = x))
  )
}

make_specs <- function(var) as.formula(paste("~", var))
lsm_by_param <- function(x, indep_var = "diabcd_n") {
  lapply(
    x,
    \(x) {
      mod_em <- emmeans::emmeans(x$lm, specs = make_specs(indep_var))
      y <- multcomp::cld(mod_em, Letters = letters)
      y <- as.data.frame(y)
      cbind(
        paramcd = unique(x$data$paramcd),
        y,
```

```
        diabcd_f = car::Anova(x$lm)[indep_var, "Pr(>F)"]
      )
    }
  )
}
```

```
tab_09_01_raw <- ads |>
  (\(x) split(x, f = x$paramcd))() |>
  lm_by_paramcd(dep_var = "chg", covariate = "base", indep_var = "diabcd_n") |>
  lsm_by_param(indep_var = "diabcd_n") |>
  (\(x) Reduce(rbind, x))()

tab_09_01 <- tab_09_01_raw |>
  (\(df) df[order(df$diabcd_n), ])() |>
  within({
    val <- paste0(
      signif(emmean, 3),
      " (", signif(lower.CL, 3), ", ", signif(upper.CL, 3), ")"
    )
    pval <- format_pval(diabcd_f)
  }) |>
  tidyr::pivot_wider(
    id_cols = c("paramcd", "pval", "diabcd_f"),
    values_from = "val",
    names_from = "diabcd_n"
  ) |>
  (\(x) x[order(x$diabcd_f), ])()
tab_09_01
```

```
library(flextable)
```

```
Attaching package: 'flextable'

The following objects are masked from 'package:latarnia.utils':

    add_footer, add_header
```

```r
wrap_line <- function(x) paste(strwrap(x, width = 80), collapse = " ")
tab_09_01_ft <- tab_09_01 |>
  subset(select = -diabcd_f) |>
  flextable() |>
  autofit() |>
  footnote(
    value = as_paragraph(
      "Note: rows are ordered by increasing p values, most significant on top."
    ),
    ref_symbols = ""
  ) |>
  footnote(
    part = "header",
    i = 1, j = 2,
    value = as_paragraph(
      "Note: pval, p value of diabetes group effect test by F test."
    ),
    ref_symbols = "a"
  ) |>
  footnote(
    value = as_paragraph(wrap_line(
      "Source: Full Analysis Set, observed cases at baseline and post
      intervention."
    )),
    ref_symbols = ""
  ) |>
  add_header_lines(wrap_long_lines(
    "Analysis Set: Full Analysis Set - Observed Cases"
  )) |>
  set_caption(
    caption =  wrap_long_lines(
      "Tab 09 01 - Analysis of Covariance / Least Means Square estimations of
      Anthropometrics Changes from baseline by Diabetes Group at
      Month 3 (95% Confidence Interval) Adjusted for Baseline"
    )
  ) |>
  theme_booktabs()
tab_09_01_ft
```

```
Warning: Warning: fonts used in `flextable` are ignored because the `pdflatex`
engine is used and not `xelatex` or `lualatex`. You can avoid this warning
```

by using the `set_flextable_defaults(fonts_ignore=TRUE)` command or use a compatible engine by defining `latex_engine: xelatex` in the YAML header of the R Markdown document.

```
bnm <- "tab_09_01"
dir_tab <- params$paths$tab
dir_dta <- params$paths$dta

file.path(dir_tab, paste0(bnm, "_ft.RData")) %T>%
  message("[output] Table saved as ", .) %>%
  save(tab_09_01_ft, file = .)
```

```
[output] Table saved as ../tlg/tables/tab_09_01_ft.RData
```

```
file.path(dir_dta, paste0(bnm, ".RData")) %T>%
  message("[output] Table saved as ", .) %>%
  save(tab_09_01, file = .)
```

```
[output] Table saved as ../data/tab_09_01.RData
```

```
file.path(dir_tab, paste(bnm, sep = ".", "docx")) %T>%
  message("[output] Table saved as ", .) %>%
  save_as_docx(tab_09_01_ft, path = .)
```

```
[output] Table saved as ../tlg/tables/tab_09_01.docx
```

```
file.path(dir_tab, paste(bnm, sep = ".", "html")) %T>%
  message("[output] Table saved as ", .) %>%
  save_as_html(tab_09_01_ft, path = .)
```

```
[output] Table saved as ../tlg/tables/tab_09_01.html
```

```
file.path(dir_dta, paste(bnm, sep = ".", "csv")) %T>%
  message("[output] Table saved as ", .) %>%
```

```r
    write.csv(tab_09_01, file = ., row.names = FALSE)
```

```
[output] Table saved as ../data/tab_09_01.csv
```

4 RNASeq - Refresher

4.1 RNA Seq - Local Study of Confounder Adjustment's Impact

Target:

- ☒ refresh the differential expression analysis technics with DESeq2.
- ☒ upgrade environment for differential expression analysis.
- ☒ evaluate the impact of confounder adjustment on a specific use case: compare miRNA expression between T2D and NGT at baseline.

```
params <- if (exists("params")) {
  c(params, yaml::read_yaml("_prog.yml"))
} else {
  yaml::read_yaml("_prog.yml")
}

devtools::load_all("src/pkg/dbs.data")
```

```
i Loading dbs.data
```

```
devtools::load_all("src/pkg/latarnia.utils")
```

```
i Loading latarnia.utils
```

```
Loading required package: grid
```

```
Loading required package: shiny
```

```
knitr::opts_chunk$set(results = params$knitr$results)

library(assertthat)
```

```
source("R/ngs.R")
```

4.1.1 Data preparation

```
adsl <- dbs.data::adsl
advs <- dbs.data::advs
adlb <- dbs.data::adlb

#' Subjid and Visit to Sample
#'
subjvis_to_spl <- function(df) paste0(df$subjid, "v", df$avisitn)

ads <- adlb |>
  subset(
    paramcd %in% c(
      "CHOL", "HBA1C", "HDL", "HOMAB", "HOMAIR", "LDL", "MATSUDA", "TRIG"
    )
  ) %>%
  rbind(advs) |>
  subset(select = -c(ct, dtype, param, base, basetype, chg, pchg)) |>
  tidyr::pivot_wider(names_from = "paramcd", values_from = "aval") |>
  (\(df) merge(adsl, df, by = "subjid"))() |>
  (\(df) S4Vectors::DataFrame(df, row.names = subjvis_to_spl(df)))() |>
  (\(df) {
    assertthat::assert_that(all(table(subjvis_to_spl(df)) == 1))
    df
  })()

ads
```

```
rna <- list(# There will be mi-RNA data.
  mrna = dbs.data::mrna_raw,
  premirna = dbs.data::premirna_raw,
  mirna = dbs.data::mirna_raw
)

rna[c("premirna", "mirna")] <- lapply(
  X = rna[c("premirna", "mirna")],
  FUN = format_mirna
```

```
)

# Rows represent genes.
rna <- lapply(X = rna, FUN = function(x) y <- x[rowSums(x) > 0, ])
rna <- lapply(X = rna, as.matrix)
assertthat::assert_that(all(colnames(rna$premirna) == colnames(rna$mirna)))
rna$allmirna <- rbind(rna$premirna, rna$mirna)
library(testthat)
test_that("rna features discriminated in noexpr, expr", {
  lapply(
    X = rna,
    FUN = function(x) expect_true(all(rowSums(x) > 0))
  )
})
```

```
library(MultiAssayExperiment)
```

```
Loading required package: SummarizedExperiment
```

```
Loading required package: MatrixGenerics
```

```
Loading required package: matrixStats
```

```

Attaching package: 'MatrixGenerics'
```

```
The following objects are masked from 'package:matrixStats':

    colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
    colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
    colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
    colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
    colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
    colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
    colWeightedMeans, colWeightedMedians, colWeightedSds,
    colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
    rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
    rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
    rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
```

```
    rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
    rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
    rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
    rowWeightedSds, rowWeightedVars

Loading required package: GenomicRanges

Loading required package: stats4

Loading required package: BiocGenerics


Attaching package: 'BiocGenerics'

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

    IQR, mad, sd, var, xtabs

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

    anyDuplicated, append, as.data.frame, basename, cbind, colnames,
    dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
    grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
    order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
    rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
    union, unique, unsplit, which.max, which.min

Loading required package: S4Vectors


Attaching package: 'S4Vectors'

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

    expand.grid, I, unname

Loading required package: IRanges

Loading required package: GenomeInfoDb
```

```
Loading required package: Biobase

Welcome to Bioconductor

    Vignettes contain introductory material; view with
    'browseVignettes()'. To cite Bioconductor, see
    'citation("Biobase")', and for packages 'citation("pkgname")'.


Attaching package: 'Biobase'

The following object is masked from 'package:MatrixGenerics':

    rowMedians

The following objects are masked from 'package:matrixStats':

    anyMissing, rowMedians
```

```
#' (Sample-)Map Arrays
#'
#' Use the colnames of `x` to deduce the `primary` and `colnames`.
#' This is used to generate the sample mapping between colData and Experiments.
#'
#' @param x (`dataframe`).
#'
#' @note In our case, primary and colnames are equivalent, colnames could
#' be different from primary names when a biological sample has different
#' names in the biological assays (e.g. machine constraint, technical
#' repetitions).
#'
#' @seealso [MultiAssayExperiment::listToMap()]
#' @examples
#' \dontrun{
#' lapply(rna, map_arrays)
#' MultiAssayExperiment::listToMap(lapply(rna, map_arrays))
#' }
#'
map_arrays <- function(x) {
  y <- data.frame(colname = colnames(x))
```

```
  y$primary <- y$colname
  y
}

besd_mae <- MultiAssayExperiment(
  experiments = ExperimentList(rna),
  colData = ads,
  sampleMap = listToMap(lapply(rna, map_arrays))
)

besd_mae
```

4.2 DE: Baseline, all micro RNA, no confounding factor (`dds_1`)

```
ctrl <- yaml::read_yaml("_prog.yml")$rna

ngs_assay <- "allmirna"

filter_for_depth <- function(mae, assay, depth_threshold) {
  mae[, colSums(mae[[ngs_assay]]) > depth_threshold, ]
}

filter_for_visit <- function(mae, visit) {
  mae[, colData(mae)$avisit == visit, ]
}

filter_for_low_expr <- function(mae, assay, cpm_threshold, frac_cols = 1 / 2) {
  # Genes expressed at least cpm_threshold in frac_cols columns
  mae[
    rowSums(cpm(mae[[assay]]) > cpm_threshold) >
      ncol(mae[[assay]]) * frac_cols,
    ,
  ]
}

ads <- besd_mae |>
  (\(mae) mae[ , , ngs_assay])() |>
  filter_for_depth("allmirna", ctrl$depth_threshold[[ngs_assay]]) |>
  filter_for_visit("Baseline") |>
```

```
    filter_for_low_expr("allmirna", ctrl$cpm_threshold[[ngs_assay]])
```

```
Warning: 'experiments' dropped; see 'metadata'
```

```
harmonizing input:
  removing 282 sampleMap rows not in names(experiments)
```

```
  ads

  dds_1 <- DESeq2::DESeqDataSetFromMatrix(
    countData = ads[[ngs_assay]],
    colData = colData(ads),
    design = stats::formula(~ diabcd)
  )

  dds_1_res <- DESeq2::DESeq(
    object = dds_1,
    quiet = FALSE, # default: FALSE
    minReplicatesForReplace = 7, # default: 7
    useT = FALSE, # default: FALSE
    minmu = 0.5, # default: 0.5
    parallel = TRUE,
    BPPARAM = BiocParallel::bpparam()
  )
```

```
estimating size factors
```

```
estimating dispersions
```

```
gene-wise dispersion estimates: 2 workers
```

```
mean-dispersion relationship
```

```
-- note: fitType='parametric', but the dispersion trend was not well captured by the
   function: y = a/x + b, and a local regression fit was automatically substituted.
   specify fitType='local' or 'mean' to avoid this message next time.
```

```
final dispersion estimates, fitting model and testing: 2 workers
```

```
-- replacing outliers and refitting for 13 genes
-- DESeq argument 'minReplicatesForReplace' = 7
-- original counts are preserved in counts(dds)

estimating dispersions

fitting model and testing
```

```
dds_1_de <- DESeq2::results(
  dds_1_res,
  contrast = c("diabcd", test = "T2D", ref = "NGT"),
  pAdjustMethod = ctrl$adj_meth
) |>
  (\(df) {
    df$feature <- rownames(df)
    df
  }) () |>
  within(log_padj <- -1 * log10(padj))

library(ggplot2)
dds_1_gg <-
  dds_1_de |> as.data.frame() |>
  ggplot(mapping = aes(log2FoldChange, log_padj, fill = log10(..count..))) +
  geom_hline(yintercept = -1 * log10(c(0.05, 0.001)), lty = 2, lwd = .5) +
  geom_vline(xintercept = c(-1, 1), lty = 2) +
  annotate(
    geom  = "label", x = -Inf, y = -1 * log10(0.05),
    label = "p = 0.05",
    fill = "white", hjust = "left", size = 2, alpha = 1
  ) +
  annotate(
    geom  = "label", x = -Inf, y = -1 * log10(0.001),
    label = "p = 0.001",
    fill = "white", hjust = "left", size = 2, alpha = 1
  ) +
  xlab("Log2-fold-change") +
  ylab(expression(-1 %*% log10(padj))) +
  stat_bin_hex() +
  scale_fill_gradient(low = "black", high = "gray90") +
  theme_minimal() +
  theme(legend.position = "bottom", asp = 2 / 3)
```

```
dds_1_gg
```

4.3 DE: Baseline, all micro RNA, accounting for Age, BMI, DCAL, Trainn (dds_2)

```
ctrl <- yaml::read_yaml("_prog.yml")$rna

ngs_assay <- "allmirna"

filter_for_depth <- function(mae, assay, depth_threshold) {
  mae[, colSums(mae[[ngs_assay]]) > depth_threshold, ]
}

filter_for_visit <- function(mae, visit) {
  mae[, colData(mae)$avisit == visit, ]
}

filter_for_low_expr <- function(mae, assay, cpm_threshold, frac_cols = 1 / 2) {
  # Genes expressed at least cpm_threshold in frac_cols columns
  mae[
    rowSums(cpm(mae[[assay]]) > cpm_threshold) >
      ncol(mae[[assay]]) * frac_cols,
    ,
  ]
}

scale_confounder <- function(mae, confounder) {
  for (i in seq_along(confounder)) {
    cfd <- confounder[i]
    colData(mae)[cfd] <- scale(colData(mae)[cfd])
  }
  mae
}
ads <- besd_mae |>
  (\(mae) mae[ , , ngs_assay])() |>
  filter_for_depth("allmirna", ctrl$depth_threshold[[ngs_assay]]) |>
  filter_for_visit("Baseline") |>
  filter_for_low_expr("allmirna", ctrl$cpm_threshold[[ngs_assay]]) |>
```

```
    scale_confounder(confounder = c("age", "trainn", "BMI", "DCAL"))
```

```
Warning: 'experiments' dropped; see 'metadata'
```

```
harmonizing input:
  removing 282 sampleMap rows not in names(experiments)
```

```
  ads

  dds_2 <- DESeq2::DESeqDataSetFromMatrix(
    countData = ads[[ngs_assay]],
    colData = colData(ads),
    design = stats::formula(~ age + BMI + trainn + DCAL + diabcd)
  )

  dds_2_res <- DESeq2::DESeq(
    object = dds_2,
    quiet = FALSE, # default: FALSE
    minReplicatesForReplace = 7, # default: 7
    useT = FALSE, # default: FALSE
    minmu = 0.5, # default: 0.5
    parallel = TRUE,
    BPPARAM = BiocParallel::bpparam()
  )
```

```
estimating size factors

estimating dispersions

gene-wise dispersion estimates: 2 workers

mean-dispersion relationship

-- note: fitType='parametric', but the dispersion trend was not well captured by the
   function: y = a/x + b, and a local regression fit was automatically substituted.
   specify fitType='local' or 'mean' to avoid this message next time.

final dispersion estimates, fitting model and testing: 2 workers
```

```r
dds_2_de <- DESeq2::results(
  dds_2_res,
  contrast = c("diabcd", test = "T2D", ref = "NGT"),
  pAdjustMethod = ctrl$adj_meth
) |>
  (\(df) {
    df$feature <- rownames(df)
    df
  }) () |>
  within(log_padj <- -1 * log10(padj))

library(ggplot2)
dds_2_gg <-
  dds_2_de |> as.data.frame() |>
  ggplot(mapping = aes(log2FoldChange, log_padj, fill = log10(..count..))) +
  geom_hline(yintercept = -1 * log10(c(0.05, 0.001)), lty = 2, lwd = .5) +
  geom_vline(xintercept = c(-1, 1), lty = 2) +
  annotate(
    geom  = "label", x = -Inf, y = -1 * log10(0.05),
    label = "p = 0.05",
    fill = "white", hjust = "left", size = 2, alpha = 1
  ) +
  annotate(
    geom  = "label", x = -Inf, y = -1 * log10(0.001),
    label = "p = 0.001",
    fill = "white", hjust = "left", size = 2, alpha = 1
  ) +
  xlab("Log2-fold-change") +
  ylab(expression(-1 %*% log10(padj))) +
  stat_bin_hex() +
  scale_fill_gradient(low = "black", high = "gray90") +
  theme_minimal() +
  theme(legend.position = "bottom", asp = 2 / 3)
```

```r
dds_2_gg
```

4.4 Comparison with/without confounding factors

```
theme_fun <- function(...) {
  theme_minimal() +
    theme(
      title = element_text(size = 9),
      text = element_text(size = 9)
    ) +
    theme(...)
}

gg_1_2 <- rbind(
  within(as.data.frame(dds_1_de), facet <- "No confounding factors"),
  within(as.data.frame(dds_2_de), facet <- "~Age + BMI + DCAL + Train")
) |>
  ggplot(mapping = aes(log2FoldChange, log_padj, fill = log10(..count..))) +
  geom_hline(yintercept = -1 * log10(c(0.05, 0.001)), lty = 2, lwd = .5) +
  geom_vline(xintercept = c(-1, 1), lty = 2) +
  annotate(
    geom  = "label", x = -Inf, y = -1 * log10(0.05),
    label = "p = 0.05",
    fill = "white", hjust = "left", size = 2, alpha = 1
  ) +
  annotate(
    geom  = "label", x = -Inf, y = -1 * log10(0.001),
    label = "p = 0.001",
    fill = "white", hjust = "left", size = 2, alpha = 1
  ) +
  xlab("Log2-fold-change") +
  ylab(expression(-1 %*% log10(padj))) +
  stat_bin_hex() +
  scale_fill_gradient(low = "black", high = "gray90") +
  facet_wrap(facet ~ ., ncol = 2) +
  theme_fun(
    legend.position = "bottom"
  ) +
  theme(
    legend.key.width = unit(5, "lines"),
    legend.key.height = unit(.8, "lines")
  )
```

```r
res <- merge(
  as.data.frame(dds_1_de),
  as.data.frame(dds_2_de),
  by = "feature",
  all = TRUE,
  suffixes = c(".asis", ".cfd")
)


fun_label <- function(df,
                      x = "log2FoldChange.asis",
                      y = "log2FoldChange.cfd") {
  cor_fun <- function(meth = "pearson") {
    round(cor(df[[x]], df[[y]], method = meth), 2)
  }
  paste0(
    "atop(",
    "r == ", cor_fun(), ",",
    "rho == ", cor_fun("spearman"),
    ")"
  )
}

lim <- range(unlist(res[c("log2FoldChange.asis", "log2FoldChange.cfd")]))
gg_cor_lfc <- ggplot(res, aes(log2FoldChange.asis, log2FoldChange.cfd)) +
  geom_hex() +
  scale_fill_viridis_c(option = "F", begin = .1, end = .9) +
  geom_abline(slope = 1, intercept = 0, col = "red") +
  annotate(
    "label", x = -Inf, y = Inf, hjust = 0, vjust = 1,
    label = fun_label(res),
    parse = TRUE,
    family = "mono",
    size = 3
  )+
  coord_cartesian(xlim = lim, ylim = lim) +
  labs(
    title = "Log Fold Change (LFC)",
    subtitle = "With / Without Adjustment for Confounding Factors",
    x = "No Adjustment",
    y = "Adjusted for Confounding Factors"
```

```
  ) +
  theme_fun(asp = 1)

lim <- range(unlist(res[c("log_padj.asis", "log_padj.cfd")]))
gg_cor_pval <- ggplot(res, aes(log_padj.asis, log_padj.cfd)) +
  geom_hex() +
  scale_fill_viridis_c(option = "D", begin = .1, end = .9) +
  geom_abline(slope = 1, intercept = 0, col = "green2") +
  annotate(
    "label", x = -Inf, y = Inf, hjust = 0, vjust = 1,
    label = fun_label(res, "log_padj.asis", "log_padj.cfd"),
    parse = TRUE,
    family = "mono",
    size = 3
  )+
  coord_cartesian(xlim = lim, ylim = lim, clip = "off") +
  labs(
    title = expression("Significance: "*-1 %.% log10(padj)),
    subtitle = "With / Without Adjustment for Confounding Factors",
    x = "No Adjustment",
    y = "Adjusted for Confounding Factors"
  ) +
  theme_fun(asp = 1)

library(cowplot)
p <- plot_grid(
  plot_grid(gg_1_2) + theme(plot.background = element_rect(color = "black")),
  plot_grid(
    plot_grid(gg_cor_lfc) +
      theme(plot.background = element_rect(color = "black")),
    plot_grid(gg_cor_pval) +
      theme(plot.background = element_rect(color = "black")),
    labels = c("B", "C")
  ),
  ncol = 1, rel_heights = c(3, 2),
  labels = c("A", NA)
)


p <- clean_slate() |>
  add_header(c("FCA Collin", "UMB BESD"), c("Confidential", "Draft")) |>
```

```
  add_title(
    c(
      "Figure 1",
      strwrap(
        "Volcano plot - Level and significance of Differential Expression among
        all miRNA at Baseline between T2D and NGT", width = 80
      ),
      "Analysis Set: Full Analysis Set"
    )
  ) |>
  add_note(c(
    "A: Left panel accounts for Age, BMI, DCAL (diet) and number of trainings in
    the estimation and test of the differential expression of every gene; it
    may present marginal differences with the version presented 2 years ago
    likely due to slight variations in stochastic elements (e.g. missing
    data imputation).",
    "A: Right panel discards any confounding factors.",
    "B, C: Scatter plots comparing
    the Log Fold Change estimations (B)/
    the significance (C, the higher the more significant)
    with (y axis) without (x axis) adjustment for confounding factors with
    annotation corresponding to the Pearson's correlation (r) and
    Spearman's rank correlation (rho).",
    "Hexbin representation: the intensity of each hexagonal bin accounts for
    the number of genes found in the area it covers."
  )) |>
  add_figure(p, height = .9) |>
  add_footer(
    "Program t2d_06_rna / Env ayup_dbs:v0.1.0-alpha",
    params$version
  )
```

```
Warning: Removed 1 rows containing missing values (geom_text).
```

```
export_as(
  p,
  file = file.path(params$paths$grh, "fig_06_01.pdf"),
  file_graph_alone = file.path(params$paths$grh, "fig_06_01_af.pdf")
)
```

```
[log] output saved as: ../tlg/graph/fig_06_01.pdf

[log] output saved as: ../tlg/graph/fig_06_01_af.pdf (annot. free)
```

```
show_slate(p)
```

5 micro RNA

5.1 miRNA Seq - Differential expression analysis

Target:

- ☒ miRNA DE at baseline without confounding factors.

```
params <- if (exists("params")) {
  c(params, yaml::read_yaml("_prog.yml"))
} else {
  yaml::read_yaml("_prog.yml")
}

devtools::load_all("src/pkg/dbs.data")
```

```
i Loading dbs.data
```

```
devtools::load_all("src/pkg/latarnia.utils")
```

```
i Loading latarnia.utils

Loading required package: grid

Loading required package: shiny
```

```
knitr::opts_chunk$set(results = params$knitr$results)
library(assertthat)
source("R/ngs.R")
```

5.1.1 Data preparation

```r
adsl <- dbs.data::adsl
advs <- dbs.data::advs
adlb <- dbs.data::adlb

#' Subjid and Visit to Sample
#'
subjvis_to_spl <- function(df) paste0(df$subjid, "v", df$avisitn)

ads <- adlb |>
  subset(
    paramcd %in% c(
      "CHOL", "HBA1C", "HDL", "HOMAB", "HOMAIR", "LDL", "MATSUDA", "TRIG"
    )
  ) %>%
  rbind(advs) |>
  subset(select = -c(ct, dtype, param, base, basetype, chg, pchg)) |>
  tidyr::pivot_wider(names_from = "paramcd", values_from = "aval") |>
  (\(df) merge(adsl, df, by = "subjid"))() |>
  (\(df) S4Vectors::DataFrame(df, row.names = subjvis_to_spl(df)))() |>
  (\(df) {
    assertthat::assert_that(all(table(subjvis_to_spl(df)) == 1))
    df
  })()

ads
```

```r
rna <- list(
  mrna = dbs.data::mrna_raw,
  premirna = dbs.data::premirna_raw,
  mirna = dbs.data::mirna_raw
)

rna[c("premirna", "mirna")] <- lapply(
  X = rna[c("premirna", "mirna")],
  FUN = format_mirna
)

# Rows represent genes.
rna <- lapply(X = rna, FUN = function(x) y <- x[rowSums(x) > 0, ])
rna <- lapply(X = rna, as.matrix)
assertthat::assert_that(all(colnames(rna$premirna) == colnames(rna$mirna)))
```

```
rna$allmirna <- rbind(rna$premirna, rna$mirna)
library(testthat)
test_that("rna features discriminated in noexpr, expr", {
  lapply(
    X = rna,
    FUN = function(x) expect_true(all(rowSums(x) > 0))
  )
})
```

```
library(MultiAssayExperiment)
```

```
Loading required package: SummarizedExperiment
```

```
Loading required package: MatrixGenerics
```

```
Loading required package: matrixStats
```

```

Attaching package: 'MatrixGenerics'
```

```
The following objects are masked from 'package:matrixStats':

    colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
    colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
    colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
    colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
    colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
    colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
    colWeightedMeans, colWeightedMedians, colWeightedSds,
    colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
    rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
    rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
    rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
    rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
    rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
    rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
    rowWeightedSds, rowWeightedVars
```

```
Loading required package: GenomicRanges
```

```
Loading required package: stats4

Loading required package: BiocGenerics


Attaching package: 'BiocGenerics'

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

    IQR, mad, sd, var, xtabs

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

    anyDuplicated, append, as.data.frame, basename, cbind, colnames,
    dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
    grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
    order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
    rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
    union, unique, unsplit, which.max, which.min

Loading required package: S4Vectors


Attaching package: 'S4Vectors'

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

    expand.grid, I, unname

Loading required package: IRanges

Loading required package: GenomeInfoDb

Loading required package: Biobase

Welcome to Bioconductor

    Vignettes contain introductory material; view with
    'browseVignettes()'. To cite Bioconductor, see
    'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
Attaching package: 'Biobase'

The following object is masked from 'package:MatrixGenerics':

    rowMedians

The following objects are masked from 'package:matrixStats':

    anyMissing, rowMedians
```

```
#' (Sample-)Map Arrays
#'
#' Use the colnames of `x` to deduce the `primary` and `colnames`.
#' This is used to generate the sample mapping between colData and Experiments.
#'
#' @param x (`dataframe`).
#'
#' @note In our case, primary and colnames are equivalent, colnames could
#' be different from primary names when a biological sample has different
#' names in the biological assays (e.g. machine constraint, technical
#' repetitions).
#'
#' @seealso [MultiAssayExperiment::listToMap()]
#' @examples
#' \dontrun{
#' lapply(rna, map_arrays)
#' MultiAssayExperiment::listToMap(lapply(rna, map_arrays))
#' }
#'
map_arrays <- function(x) {
  y <- data.frame(colname = colnames(x))
  y$primary <- y$colname
  y
}

besd_mae <- MultiAssayExperiment(
  experiments = ExperimentList(rna),
  colData = ads,
  sampleMap = listToMap(lapply(rna, map_arrays))
)
```

```
besd_mae
```

5.2 `dds_1` - DE: Baseline, all micro RNA

```
ctrl <- yaml::read_yaml("_prog.yml")$rna

ngs_assay <- "allmirna"

filter_for_depth <- function(mae, assay, depth_threshold) {
  mae[, colSums(mae[[ngs_assay]]) > depth_threshold, ]
}

filter_for_visit <- function(mae, visit) {
  mae[, colData(mae)$avisit == visit, ]
}

filter_for_low_expr <- function(mae, assay, cpm_threshold, frac_cols = 1 / 2) {
  # Genes expressed at least cpm_threshold in frac_cols columns
  mae[
    rowSums(cpm(mae[[assay]]) > cpm_threshold) >
      ncol(mae[[assay]]) * frac_cols,
    ,
  ]
}

ads <- besd_mae |>
  (\(mae) mae[ , , ngs_assay])() |>
  filter_for_depth("allmirna", ctrl$depth_threshold[[ngs_assay]]) |>
  filter_for_visit("Baseline") |>
  filter_for_low_expr("allmirna", ctrl$cpm_threshold[[ngs_assay]])
```

```
Warning: 'experiments' dropped; see 'metadata'
```

```
harmonizing input:
  removing 282 sampleMap rows not in names(experiments)
```

```r
dds_1_dta <- DESeq2::DESeqDataSetFromMatrix(
  countData = ads[[ngs_assay]],
  colData = colData(ads),
  design = stats::formula(~ diabcd)
)
```

```r
dds_1_fit <- DESeq2::DESeq(
  object = dds_1_dta,
  quiet = FALSE, # default: FALSE
  minReplicatesForReplace = 7, # default: 7
  useT = FALSE, # default: FALSE
  minmu = 0.5, # default: 0.5
  parallel = TRUE,
  BPPARAM = BiocParallel::bpparam()
)
```

```
estimating size factors

estimating dispersions

gene-wise dispersion estimates: 2 workers

mean-dispersion relationship

-- note: fitType='parametric', but the dispersion trend was not well captured by the
   function: y = a/x + b, and a local regression fit was automatically substituted.
   specify fitType='local' or 'mean' to avoid this message next time.

final dispersion estimates, fitting model and testing: 2 workers

-- replacing outliers and refitting for 13 genes
-- DESeq argument 'minReplicatesForReplace' = 7
-- original counts are preserved in counts(dds)

estimating dispersions

fitting model and testing
```

```
de_by_ctrs <- function(df,
                       ctrs,
                       adj_meth = ctrl$adj_meth) {
  lapply(
    ctrs,
    fit = df,
    adj_meth = adj_meth,
    FUN = function(x, fit, adj_meth) {
      y <- DESeq2::results(fit, contrast = x, pAdjustMethod = adj_meth)
      y$feature <- rownames(y)
      y$log_padj <- -1 * log10(y$padj)
      y$ctrs <- paste(x["test"], "vs", x["ref"])
      as.data.frame(y)
    }
  )
}
dds_1_est <-
  dds_1_fit |>
  de_by_ctrs(
    ctrs = list(
      c("diabcd", test = "T2D", ref = "NGT"),
      c("diabcd", test = "T2D", ref = "IGT"),
      c("diabcd", test = "IGT", ref = "NGT")
    )
  ) |>
  (\(x) Reduce(rbind, x))()
```

```
library(ggplot2)
dds_1_gg <-
  dds_1_est |>
  ggplot(mapping = aes(log2FoldChange, log_padj, fill = log10(..count..))) +
  geom_hline(yintercept = -1 * log10(c(0.05, 0.001)), lty = 2, lwd = .5) +
  geom_vline(xintercept = c(-1, 1), lty = 2) +
  annotate(
    geom  = "label", x = -Inf, y = -1 * log10(0.05),
    label = "p = 0.05",
    fill = "white", hjust = "left", size = 2, alpha = 1
  ) +
  annotate(
    geom  = "label", x = -Inf, y = -1 * log10(0.001),
    label = "p = 0.001",
```

```
    fill = "white", hjust = "left", size = 2, alpha = 1
  ) +
  xlab("Log2-fold-change") +
  ylab(expression(-1 %*% log10(padj))) +
  stat_bin_hex() +
  scale_fill_gradient(low = "black", high = "gray90") +
    facet_grid(. ~ ctrs) +
  theme_minimal() +
  theme(legend.position = "bottom", asp = 2 / 3)
```

```
p <- clean_slate() |>
  add_header(c("FCA Collin", "UMB BESD"), c("Confidential", "Draft")) |>
  add_title(
    c(
      "Figure 1",
      strwrap(
        "Volcano plot - Response Size and Significance of Differential
        Expression among all miRNA at Baseline", width = 80
      ),
      "Analysis Set: Full Analysis Set"
    )
  ) |>
  add_figure(dds_1_gg, height = .9) |>
  add_footer(
    "Program t2d_07_mir / Env ayup_dbs:v0.1.0-alpha",
    params$version
  )
```

```
export_as(
  p,
  file = file.path(params$paths$grh, "fig_07_01.pdf"),
  file_graph_alone = file.path(params$paths$grh, "fig_07_01_af.pdf")
)
```

```
[log] output saved as: ../tlg/graph/fig_07_01.pdf
```

```
[log] output saved as: ../tlg/graph/fig_07_01_af.pdf (annot. free)
```

```
show_slate(p)
```

References

Committee for Medicinal Products for Human Use (CHMP). 2015. "Guideline on Adjustment for Baseline Covariates in Clinical Trials." European Medicines Agency. https://www.ema.europa.eu/en/documents/scientific-guideline/guideline-adjustment-baseline-covariates-clinical-trials_en.pdf.

COMMITTEE FOR PROPRIETARY MEDICINAL PRODUCTS. 2003. "POINTS TO CONSIDER ON ADJUSTMENT FOR BASELINE COVARIATES." The European Agency for the Evaluation of Medicinal Products Evaluation of Medicines for Human Use. https://www.ema.europa.eu/en/documents/scientific-guideline/points-consider-adjustment-baseline-covariates_en.pdf.

Liu, Guanghan F, Kaifeng Lu, Robin Mogg, Madhuja Mallick, and Devan V Mehrotra. 2009. "Should Baseline Be a Covariate or Dependent Variable in Analyses of Change from Baseline in Clinical Trials?" *Statistics in Medicine* 28 (20): 2509–30. https://doi.org/10.1002/sim.3639.

O'Connell, Nathaniel S, Lin Dai, Yunyun Jiang, Jaime L Speiser, Ralph Ward, Wei Wei, Rachel Carroll, and Mulugeta Gebregziabher. 2017. "Methods for Analysis of Pre-Post Data in Clinical Research: A Comparison of Five Common Methods." *Journal of Biometrics & Biostatistics* 8 (1): 1. https://doi.org/10.4172/2155-6180.1000334.

Van Breukelen, G. 2006. "ANCOVA Versus Change from Baseline: More Power Inrandomized Studies, More Bias in Nonrandomized Studies." *Journal of ClinicalEpidemiology*, 59–920. https://doi.org/10.1016/j.jclinepi.2006.02.007.

Vickers, Andrew J. 2001. "The Use of Percentage Change from Baseline as an Outcome in a Controlled Trial Is Statistically Inefficient: A Simulation Study." *BMC Medical Research Methodology* 1 (1): 1–4. https://doi.org/10.1186/1471-2288-1-6.

Zhang, Shiyuan, James Paul, Manyat Nantha-Aree, Norman Buckley, Uswa Shahzad, Ji Cheng, Justin DeBeer, et al. 2014. "Empirical Comparison of Four Baseline Covariate Adjustment Methods in Analysis of Continuous Outcomes in Randomized Controlled Trials." *Clinical Epidemiology* 6: 227. https://doi.org/10.2147/clep.s56554.