Package: tern.mmrm (via r-universe)

September 23, 2024

Title Tables and Graphs for Mixed Models for Repeated Measures (MMRM)

Version 0.3.2

Date 2024-09-23

Description Mixed models for repeated measures (MMRM) are a popular choice for analyzing longitudinal continuous outcomes in randomized clinical trials and beyond; see for example Cnaan, Laird and Slasor (1997) <doi:10.1002/(SICI)1097-0258(19971030)16:20%3C2349::AID-SIM667%3E3.0.CO;2-E>. This package provides an interface for fitting MMRM within the 'tern' <https://cran.r-project.org/package=tern> framework by Zhu et al. (2023) and tabulate results easily using 'rtables' <https://cran.r-project.org/package=rtables> by Becker et al. (2023). It builds on 'mmrm' <https://cran.r-project.org/package=mmrm> by Sabanés Bové et al. (2023) for the actual MMRM computations.

License Apache License 2.0

URL https://github.com/insightsengineering/tern.mmrm, https://insightsengineering.github.io/tern.mmrm/

BugReports https://github.com/insightsengineering/tern.mmrm/issues

Depends R (>= 3.6), tern (>= 0.9.5)

Imports checkmate (>= 2.1.0), cowplot, dplyr (>= 1.1.0), emmeans (>= 1.10.4), formatters (>= 0.5.9), generics, ggplot2, lifecycle (>= 0.2.0), magrittr, mmrm (>= 0.3.5), parallelly (>= 1.25.0), rlang (>= 1.0.1), rtables (>= 0.6.10), stats, tidyr (>= 0.8.3)

Suggests broom (>= 0.7.10), grid, knitr (>= 1.42), maditr (>= 0.8.1), Matrix, rmarkdown (>= 2.23), testthat (>= 3.1), vdiffr (>= 1.0.7), withr (>= 2.0.0)

VignetteBuilder knitr, rmarkdown

Config/Needs/verdepcheck insightsengineering/tern, mllg/checkmate, wilkelab/cowplot, tidyverse/dplyr, rvlenth/emmeans, insightsengineering/formatters, r-lib/generics, tidyverse/ggplot2, r-lib/lifecycle, tidyverse/magrittr,

openpharma/mmrm, HenrikBengtsson/parallelly, r-lib/rlang, insightsengineering/rtables, tidyverse/tidyr, tidymodels/broom, yihui/knitr, gdemin/maditr, cran/Matrix, rstudio/rmarkdown, r-lib/testthat, r-lib/vdiffr, r-lib/withr

Config/testthat/edition 3

Encoding UTF-8

Language en-US

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Collate 'utils.R' 'assert_data.R' 'covariance_plot.R' 'fit_mmrm.R' 'formula.R' 'g_mmrm.R' 'labels.R' 'lsmeans.R' 'subgroups.R' 'tabulate_mmrm.R' 'tern.mmrm-package.R'

Repository https://insightsengineering.r-universe.dev

RemoteUrl https://github.com/insightsengineering/tern.mmrm

RemoteRef v0.3.2

RemoteSha a3b28ee2cfb354d507970222d62809157acc4813

Contents

`build_formula` *Building Model Formula*

Description

[Stable]

This builds the model formula which is used inside `fit_mmrm()` and provided to `mmrm::mmrm()` internally. It can be instructive to look at the resulting formula directly sometimes.

Usage

```
build_formula(
  vars,
 cor_struct = c("unstructured", "toeplitz", "heterogeneous toeplitz", "ante-dependence",
  "heterogeneous ante-dependence", "auto-regressive", "heterogeneous auto-regressive",
    "compound symmetry", "heterogeneous compound symmetry")
)
```

Arguments

vars	(list) variables to use in the model.
cor_struct	(string) specify the covariance structure to use.

Value

Formula to use in `mmrm::mmrm()`.

Examples

```
vars <- list(
  response = "AVAL", covariates = c("RACE", "SEX"),
  id = "USUBJID", arm = "ARMCD", visit = "AVISIT"
)
build_formula(vars, "auto-regressive")
build_formula(vars)
```

---

extract_mmrm_subgroups

Extraction of MMRM Subgroup Results based on Population Model Definition

---

Description

[Experimental]

This prepares LS mean estimates and contrasts for a specific visit and treatment arm relative to the reference arm, along with a list of subgroup variables and corresponding (grouped) factor levels.

Usage

```
extract_mmrm_subgroups(
  fit,
  visit,
  subgroups = NULL,
  groups_lists = list(),
  treatment_arm = fit$treatment_levels[1L],
  label_all = "All Patients"
)
```

Arguments

`fit`	(`tern_mmrm`) model fit on the total population.
`visit`	(`string`) single visit or name of averages of visits (referring to the averages specified when creating the `fit`).
`subgroups`	(`character` or `NULL`) names of subgroup variables to use in the forest plot, these need to be factors.
`groups_lists`	(named `list` of `list`) optionally contains for each `subgroups` variable a list, which specifies groups of factor levels, see details.
`treatment_arm`	(`string`) single treatment arm to compare with the reference arm.
`label_all`	(`string`) label for the total population analysis.

Details

The `groups_lists` argument is handy when you don't want to have subgroups identical to the original levels of the factor variable. This might be the case when you want to merge levels into a single subgroup, define overlapping subgroups or omit levels completely. Then you insert an element into `groups_lists` with the name of the `subgroups` variable and containing as a named list the subgroup definitions. See the example below.

Value

A list with two elements:

- `estimates`: `data.frame` with columns `arm`, `n`, `lsmean`, `subgroup`, `var`, `var_label`, `row_type`, containing the LS means results for the overall population and the specified subgroups.
- `contrasts`: `data.frame` with columns `n_tot`, `diff`, `lcl`, `ucl`, `pval`, `subgroup`, `var`, `var_label`, `row_type`. Note that this has half the number of rows as `estimates`.

Note

If the original model `vars` include `covariates` which are used here in `subgroups` then these are dropped from `covariates` before the corresponding model is fitted.

Examples

```
mmrm_results <- fit_mmrm(
  vars = list(
    response = "FEV1",
    covariates = "RACE",
    id = "USUBJID",
    arm = "ARMCD",
    visit = "AVISIT"
  ),
```

```
  data = mmrm_test_data,
  cor_struct = "compound symmetry",
  weights_emmeans = "equal",
  averages_emmeans = list(
    "VIS1+2" = c("VIS1", "VIS2")
  )
)

extract_mmrm_subgroups(
  fit = mmrm_results,
  visit = "VIS3",
  subgroups = c("RACE", "SEX"),
  groups_lists = list(
    RACE = list(
      A = c("Asian", "White"),
      B = c("Black or African American", "White")
    )
  )
)
```

`fit_mmrm` MMRM *Analysis*

Description

[Stable]

Does the MMRM analysis. Multiple other functions can be called on the result to produce tables and graphs.

Usage

```
fit_mmrm(
 vars = list(response = "AVAL", covariates = c(), id = "USUBJID", arm = "ARM", visit =
    "AVISIT"),
  data,
  conf_level = 0.95,
  cor_struct = "unstructured",
  weights_emmeans = "proportional",
  averages_emmeans = list(),
  parallel = FALSE,
  ...
)
```

Arguments

vars	(named list of string or character) specifying the variables in the MMRM. The following elements need to be included as character vectors and match corresponding columns in data:

- response: the response variable.
- covariates: the additional covariate terms (might also include interactions).
- id: the subject ID variable.
- arm: the treatment group variable (factor).
- visit: the visit variable (factor).
- weights: optional weights variable (if NULL or omitted then no weights will be used).

Note that the main effects and interaction of arm and visit are by default included in the model.

data (data.frame)
with all the variables specified in vars. Records with missing values in any independent variables will be excluded.

conf_level (proportion)
confidence level of the interval.

cor_struct (string)
specifying the covariance structure, defaults to "unstructured". See the details.

weights_emmeans
(string)
argument from emmeans::emmeans(), "proportional" by default.

averages_emmeans
(list)
optional named list of visit levels which should be averaged and reported along side the single visits.

parallel (flag)
controls whether the optimizer search can use available free cores on the machine (not default).

... additional arguments for mmrm::mmrm(), in particular reml and options listed in mmrm::mmrm_control().

Details

Multiple different degree of freedom adjustments are available via the method argument for mmrm::mmrm(). In addition, covariance matrix adjustments are available via vcov. Please see mmrm::mmrm_control() for details and additional useful options.

For the covariance structure (cor_struct), the user can choose among the following options.

- unstructured: Unstructured covariance matrix. This is the most flexible choice and default. If there are T visits, then T * (T+1) / 2 variance parameters are used.
- toeplitz: Homogeneous Toeplitz covariance matrix, which uses T variance parameters.
- heterogeneous toeplitz: Heterogeneous Toeplitz covariance matrix, which uses 2 * T - 1 variance parameters.
- ante-dependence: Homogeneous Ante-Dependence covariance matrix, which uses T variance parameters.

- heterogeneous ante-dependence: Heterogeneous Ante-Dependence covariance matrix, which uses 2 * T - 1 variance parameters.
- auto-regressive: Homogeneous Auto-Regressive (order 1) covariance matrix, which uses 2 variance parameters.
- heterogeneous auto-regressive: Heterogeneous Auto-Regressive (order 1) covariance matrix, which uses T + 1 variance parameters.
- compound symmetry: Homogeneous Compound Symmetry covariance matrix, which uses 2 variance parameters.
- heterogeneous compound symmetry: Heterogeneous Compound Symmetry covariance matrix, which uses T + 1 variance parameters.

Value

A tern_mmrm object which is a list with MMRM results:

- fit: The mmrm object which was fitted to the data. Note that via mmrm::component(fit, "optimizer") the finally used optimization algorithm can be obtained, which can be useful for refitting the model later on.
- cov_estimate: The matrix with the covariance matrix estimate.
- diagnostics: A list with model diagnostic statistics (REML criterion, AIC, corrected AIC, BIC).
- lsmeans: This is a list with data frames estimates and contrasts. The attributes averages and weights save the settings used (averages_emmeans and weights_emmeans).
- vars: The variable list.
- labels: Corresponding list with variable labels extracted from data.
- cor_struct: input.
- parallel: input.
- ref_level: The reference level for the arm variable, which is always the first level.
- treatment_levels: The treatment levels for the arm variable.
- conf_level: The confidence level which was used to construct the lsmeans confidence intervals.
- additional: List with any additional inputs passed via ...

Examples

```
library(dplyr)
library(rtables)

mmrm_results <- fit_mmrm(
  vars = list(
    response = "FEV1",
    covariates = c("RACE", "SEX"),
    id = "USUBJID",
    arm = "ARMCD",
    visit = "AVISIT"
  ),
```

```
  data = mmrm_test_data,
  cor_struct = "unstructured",
  weights_emmeans = "equal",
  averages_emmeans = list(
    "VIS1+2" = c("VIS1", "VIS2")
  )
)
```

`get_mmrm_lsmeans` *Extract Least Square Means from* `MMRM`

Description

[Stable]

Extracts the least square means from an `MMRM` fit.

Usage

```
get_mmrm_lsmeans(fit, vars, conf_level, weights, averages = list())
```

Arguments

`fit`	(`mmrm`) result of `mmrm::mmrm()`.
`vars`	(named `list` of `string` or `character`) specifying the variables in the `MMRM`. The following elements need to be included as character vectors and match corresponding columns in `data`: • `response`: the response variable. • `covariates`: the additional covariate terms (might also include interactions). • `id`: the subject ID variable. • `arm`: the treatment group variable (factor). • `visit`: the visit variable (factor). • `weights`: optional weights variable (if `NULL` or omitted then no weights will be used). Note that the main effects and interaction of `arm` and `visit` are by default included in the model.
`conf_level`	(`proportion`) confidence level of the interval.
`weights`	(`string`) type of weights to be used for the least square means, see `emmeans::emmeans()` for details.
`averages`	(`list`) named list of visit levels which should be averaged and reported along side the single visits.

Value

A list with data frames `estimates` and `contrasts`. The attributes `averages` and `weights` save the settings used.

`g_covariance` *Visualization of Covariance Matrix*

Description

[Experimental]

Plot of covariance (or correlation) matrix as a function of lag or time. The covariance structure is vectorized internally and lag or time distances are computed and can be used for visualization.

Usage

```
g_covariance(
  vcov_matrix,
  time_prefix = NULL,
  x_var = c("lag", "time_diff"),
  xlab = NULL,
  ylab = ""
)
```

Arguments

`vcov_matrix`	(`matrix`) symmetric covariance matrix with identical row and column names.
`time_prefix`	(`string`) string in the names of `vcov_matrix` that precedes the time point value.
`x_var`	(`string`) can be `lag` or `time_diff` for lag or time difference, respectively.
`xlab`	(`string` or `NULL`) x-axis label, if `NULL` then automatically determined from `x_var`.
`ylab`	(`string`) y-axis label.

Details

The default `time_prefix` value is `NULL`, which assumes that the names of the input matrix don't have any character string other than time point value. If a `time_prefix` is specified, this string should appear in front of all the names in `vcov_matrix`.

Value

The `ggplot` object.

Examples

```
vcov_matrix <- matrix(
  c(49, 12, 12, 23),
  nrow = 2, ncol = 2,
  dimnames = list(
    c(1, 2),
    c(1, 2)
  )
)
g_covariance(vcov_matrix, x_var = "time_diff")
```

---

g_mmrm_diagnostic *Diagnostic Plots for* MMRM

Description

[Stable]

This function produces diagnostic plots.

Usage

```
g_mmrm_diagnostic(
  object,
  type = c("fit-residual", "q-q-residual"),
  z_threshold = NULL
)
```

Arguments

object	(tern_mmrm) model result produced by fit_mmrm().
type	(string) specifying the type of diagnostic plot to be produced: • fit-residual: A fitted vs residuals plot, grouped by visits. This allows to see if there is remaining structure in the residuals that might be captured by adding additional covariates to the model. • q-q-residual: A Q-Q plot for the residuals (i.e. sorted standardized residuals vs. normal quantiles), grouped by visits. Observations with an absolute standardized residual above z_threshold will be labeled.
z_threshold	(numeric) optional number indicating the normal quantile threshold for the Q-Q plot.

Details

Here we use marginal fitted values and residuals. That is, we estimate fitted values, and the difference of those fitted values vs. the observed data are the residuals.

Value

A ggplot2 plot.

See Also

g_mmrm_lsmeans() for plotting the least-squares means and contrasts.

Examples

```
mmrm_results <- fit_mmrm(
  vars = list(
    response = "FEV1",
    covariates = c("RACE", "SEX"),
    id = "USUBJID",
    arm = "ARMCD",
    visit = "AVISIT"
  ),
  data = mmrm_test_data,
  cor_struct = "unstructured",
  weights_emmeans = "equal"
)
g_mmrm_diagnostic(mmrm_results)
g_mmrm_diagnostic(mmrm_results, type = "q-q-residual")
```

---

g_mmrm_lsmeans *Plot LS means for MMRM*

Description

[Stable]

This function summarizes adjusted lsmeans and standard error, as well as conducts comparisons between groups' adjusted lsmeans, where the first level of the group is the reference level.

Usage

```
g_mmrm_lsmeans(
  object,
  select = c("estimates", "contrasts"),
  titles = c(estimates = paste("Adjusted mean of", object$labels$response,
    "by treatment at visits"), contrasts = paste0("Differences of ",
   object$labels$response, " adjusted means vs. control ('", object$ref_level, "')")),
  xlab = object$labels$visit,
  ylab = paste0("Estimates with ", round(object$conf_level * 100), "% CIs"),
  xlimits = NULL,
  ylimits = NULL,
  width = 0.6,
  show_pval = TRUE,
  show_lines = FALSE,
```

```
  constant_baseline = NULL,
  n_baseline = NA_integer_,
  table_stats = character(),
 table_formats = c(n = "xx.", estimate = "xx.x", se = "xx.x", ci = "(xx.xx, xx.xx)"),
  table_labels = c(n = "n", estimate = "LS mean", se = "Std. Error", ci =
    paste0(round(object$conf_level * 100), "% CI")),
  table_font_size = 3,
  table_rel_height = 0.5
)
```

Arguments

object	(tern_mmrm) model result produced by fit_mmrm().
select	(character) to select one or both of "estimates" and "contrasts" plots. Note "contrasts" option is not applicable to model summaries excluding an arm variable.
titles	(character) with elements estimates and contrasts containing the plot titles.
xlab	(string) the x axis label.
ylab	(string) the y axis label.
xlimits	(numeric) x axis limits.
ylimits	(numeric) y axis limits.
width	(numeric) width of the error bars.
show_pval	(flag) should the p-values for the differences of LS means vs. control be displayed (not default)?
show_lines	(flag) should the visit estimates be connected by lines (not default)?
constant_baseline	(named number or NULL) optional constant baseline for the LS mean estimates. If specified then needs to be a named number, and the name will be used to label the corresponding baseline visit. The differences of LS means will always be 0 at this baseline visit.
n_baseline	(int or named integer) optional number of patients at baseline. Since this can be visible in the optional table below the estimates plot, and we cannot infer this from object (since that is then only fit without baseline data), we need to allow the user to specify this. If number then assumed constant across potential treatment arms, otherwise one number per treatment arm can be provided.

table_stats (character)
: names of the statistics that will be displayed in the table below the estimates plot. Note that the table will only be added when selecting only the "estimates" plot. Available statistics are `n`, `estimate`, `se`, and `ci`.

table_formats (named character)
: format patterns for descriptive statistics used in the (optional) estimates table appended to the estimates plot.

table_labels (named character)
: labels for the statistics in the (optional) estimates table.

table_font_size (number)
: controls the font size of values in the (optional) estimates table.

table_rel_height (number)
: controls the relative height of the (optional) estimates table compared to the estimates plot.

Details

If variable labels are available in the original data set, then these are used. Otherwise the variable names themselves are used for annotating the plot.

The contrast plot is not going to be returned if treatment is not considered in the `tern_mmrm` object input, no matter if `select` argument contains the `contrasts` value.

Value

A `ggplot2` plot.

Examples

```
library(dplyr)

mmrm_results <- fit_mmrm(
  vars = list(
    response = "FEV1",
    covariates = c("RACE", "SEX"),
    id = "USUBJID",
    arm = "ARMCD",
    visit = "AVISIT"
  ),
  data = mmrm_test_data,
  cor_struct = "unstructured",
  weights_emmeans = "equal"
)
g_mmrm_lsmeans(mmrm_results, constant_baseline = c(BSL = 0))
g_mmrm_lsmeans(
  mmrm_results,
  select = "estimates",
  show_lines = TRUE,
```

```
  xlab = "Visit"
)
g_mmrm_lsmeans(
  mmrm_results,
  select = "contrasts",
  titles = c(contrasts = "Contrasts of FKSI-FWB means"),
  show_pval = TRUE,
  show_lines = TRUE,
  width = 0.8
)

mmrm_test_data2 <- mmrm_test_data %>%
  filter(ARMCD == "TRT")

mmrm_results_no_arm <- fit_mmrm(
  vars = list(
    response = "FEV1",
    covariates = c("RACE", "SEX"),
    id = "USUBJID",
    visit = "AVISIT"
  ),
  data = mmrm_test_data2,
  cor_struct = "unstructured",
  weights_emmeans = "equal"
)

g_mmrm_lsmeans(mmrm_results_no_arm, select = "estimates")
g_mmrm_lsmeans(
  mmrm_results_no_arm,
  select = c("estimates", "contrasts"),
  titles = c(
    estimates = "Adjusted mean of FKSI-FWB",
    contrasts = "it will not be created"
  ),
  show_pval = TRUE,
  width = 0.8
)

g_mmrm_lsmeans(
  mmrm_results_no_arm,
  select = "estimates",
  titles = c(estimates = "Adjusted mean of FKSI-FWB"),
  show_pval = TRUE,
  width = 0.8,
  show_lines = TRUE
)

g_mmrm_lsmeans(
  mmrm_results,
  select = "estimates",
  titles = c(estimates = "Adjusted mean of FKSI-FWB"),
  table_stats = c("n", "ci")
)
```

mmrm_test_data *Example dataset for* tern.mmrm *package.*

Description

[Stable]

Measurements of FEV1 (forced expired volume in one second) is a measure of how quickly the lungs can be emptied. Low levels of FEV1 may indicate chronic obstructive pulmonary disease (COPD).

Usage

```
mmrm_test_data
```

Format

A tibble with 800 rows and 7 variables:

- USUBJID: unique subject identifier.
- AVISIT: visit number.
- ARMCD: treatment, TRT or PBO.
- RACE: 3-category race.
- SEX: sex.
- FEV1_BL: FEV1 at baseline (%).
- FEV1: FEV1 at study visits.

tabulate_mmrm *Tabulation of* MMRM *Results*

Description

[Stable]

These functions can be used to produce tables from a fitted MMRM produced with fit_mmrm().

Usage

```
## S3 method for class 'tern_mmrm'
as.rtable(x, type = c("fixed", "cov", "diagnostic"), ...)

h_mmrm_fixed(x, format = "xx.xxxx")

h_mmrm_cov(x, format = "xx.xxxx")

h_mmrm_diagnostic(x, format = "xx.xxxx")
```

```
## S3 method for class 'tern_mmrm'
tidy(x, ...)

s_mmrm_lsmeans(df, .in_ref_col, show_relative = c("reduction", "increase"))

a_mmrm_lsmeans(df, .in_ref_col, show_relative = c("reduction", "increase"))

s_mmrm_lsmeans_single(df)

a_mmrm_lsmeans_single(df)

summarize_lsmeans(
  lyt,
  arms = TRUE,
  ...,
  table_names = "lsmeans_summary",
  .stats = NULL,
  .formats = NULL,
  .indent_mods = NULL,
  .labels = NULL
)
```

Arguments

x	(tern_mmrm) the original result from fit_mmrm().
type	(string) type of table which should be returned.
...	additional argument format for controlling the numeric format.
format	(string) format for the numbers in the table.
df	(data frame) data set containing all analysis variables.
.in_ref_col	(logical) TRUE when working with the reference level, FALSE otherwise.
show_relative	should the "reduction" (control - treatment, default) or the "increase" (treatment - control) be shown for the relative change from baseline?
lyt	(layout) input layout where analyses will be added to.
arms	(flag) should treatment variable be considered when using summarize_lsmeans layout generating function.
table_names	(character) this can be customized in case that the same vars are analyzed multiple times, to avoid warnings from rtables.

.stats	(character) statistics to select for the table.
.formats	(named character or list) formats for the statistics.
.indent_mods	(named integer) indent modifiers for the labels.
.labels	(named character) labels for the statistics (without indent).

Value

as.rtable.tern_mmrm() returns the fixed effects, covariance estimate or diagnostic statistics tables.

Functions

- as.rtable(tern_mmrm): Produce simple MMRM tables via the generic as.rtable().
- h_mmrm_fixed(): Helper function to produce fixed effects table.
- h_mmrm_cov(): Helper function to produce a covariance matrix table.
- h_mmrm_diagnostic(): Helper function to produce a diagnostic statistics table.
- tidy(tern_mmrm): Helper method (for broom::tidy()) to prepare a data.frame from an tern_mmrm object containing the least-squares means and contrasts.
- s_mmrm_lsmeans(): Statistics function which is extracting estimates from a tidied least-squares means data frame.
- a_mmrm_lsmeans(): Formatted Analysis function which can be further customized by calling rtables::make_afun() on it. It is used as afun in rtables::analyze().
- s_mmrm_lsmeans_single(): Statistics function which is extracting estimates from a tidied least-squares means data frame when ARM is not considered in the model.
- a_mmrm_lsmeans_single(): Formatted Analysis function (when ARM is not considered in the model) which can be further customized by calling rtables::make_afun() on it. It is used as afun in rtables::analyze().
- summarize_lsmeans(): Analyze function for tabulating least-squares means estimates from tidied mmrm results.

Examples

```
result <- fit_mmrm(
  vars = list(
    response = "FEV1",
    covariates = c("RACE", "SEX"),
    id = "USUBJID",
    arm = "ARMCD",
    visit = "AVISIT"
  ),
  data = mmrm_test_data,
  cor_struct = "unstructured",
```

```
  weights_emmeans = "equal"
)
as.rtable(result, type = "cov", format = "xx.x")

result_no_arm <- fit_mmrm(
  vars = list(
    response = "FEV1",
    covariates = c("RACE", "SEX"),
    id = "USUBJID",
    visit = "AVISIT"
  ),
  data = mmrm_test_data,
  cor_struct = "unstructured",
  weights_emmeans = "equal"
)
as.rtable(result_no_arm, type = "cov", format = "xx.x")
df <- broom::tidy(result)
df_no_arm <- broom::tidy(result_no_arm)
s_mmrm_lsmeans(df[8, ], .in_ref_col = FALSE)
s_mmrm_lsmeans_single(df_no_arm[4, ])
library(dplyr)

dat_adsl <- mmrm_test_data %>%
  select(USUBJID, ARMCD) %>%
  unique()
basic_table() %>%
  split_cols_by("ARMCD", ref_group = result$ref_level) %>%
  add_colcounts() %>%
  split_rows_by("AVISIT") %>%
  summarize_lsmeans(
    .stats = c("n", "adj_mean_se", "adj_mean_ci", "diff_mean_se", "diff_mean_ci"),
    .labels = c(adj_mean_se = "Adj. LS Mean (Std. Error)"),
    .formats = c(adj_mean_se = sprintf_format("%.1f (%.2f)"))
  ) %>%
  build_table(
    df = broom::tidy(result),
    alt_counts_df = dat_adsl
  )

basic_table() %>%
  split_rows_by("AVISIT") %>%
  summarize_lsmeans(arms = FALSE) %>%
  build_table(
    df = broom::tidy(result_no_arm),
    alt_counts_df = dat_adsl
  )
```

`tabulate_mmrm_subgroups`

Tabulation of MMRM Subgroups Results

Description

[Experimental]

This function tabulates the results from `extract_mmrm_subgroups()`.

Usage

```
tabulate_mmrm_subgroups(
  lyt,
  df,
  vars = c("n_tot", "n", "lsmean", "diff", "ci"),
  .formats = list(n = "xx", n_tot = "xx", lsmean = "xx.x", diff = "xx.x", ci =
    "(xx.x, xx.x)", pval = "x.xxxx | (<0.0001)"),
 .labels = list(n = "n", n_tot = "Total n", lsmean = "Mean", diff = "Mean Difference",
   ci = paste0(round(100 * df$contrasts$conf_level[1]), "% CI"), pval = "p-value")
)
```

Arguments

`lyt`	(`layout`) input layout where analyses will be added to.
`df`	(`list`) of data frames containing all analysis variables, is the result from `extract_mmrm_subgroups()`.
`vars`	(`character`) the name of statistics to be reported among `n_tot` (total number of patients per group), `n` (number of patients per treatment arm and group), `lsmean` (least square mean point estimate), `diff` (difference of least square mean estimates between treatment and reference arm), `ci` (confidence interval of difference) and `pval` (p value of the `diff`, not adjusted for multiple comparisons). Note, the statistics `n_tot`, `diff` and `ci` are required.
`.formats`	(named `list`) containing the formats for the statistics.
`.labels`	(named `list`) containing the labels for the statistics.

Value

The `rtables` object.

Examples

```
mmrm_results <- fit_mmrm(
  vars = list(
    response = "FEV1",
    covariates = "RACE",
    id = "USUBJID",
    arm = "ARMCD",
    visit = "AVISIT"
  ),
```

```
  data = mmrm_test_data,
  cor_struct = "compound symmetry",
  weights_emmeans = "equal",
  averages_emmeans = list(
    "VIS1+2" = c("VIS1", "VIS2")
  )
)

df <- extract_mmrm_subgroups(
  fit = mmrm_results,
  visit = "VIS3",
  subgroups = c("RACE", "SEX")
)

## Table with default columns.
basic_table() %>%
  tabulate_mmrm_subgroups(df)

## Table with selected columns.
tab <- basic_table() %>%
  tabulate_mmrm_subgroups(
    df = df,
    vars = c("n_tot", "diff", "ci", "pval")
  )
tab

## Forest plot can be produced based on this very easily.
g_forest(tab, logx = FALSE, xlim = c(-10, 10), x_at = c(-10, -5, 0, 5, 10), vline = 0)
```

Index