# Package ‘admiralmetabolic’

January 20, 2025

**Type** Package

**Title** Metabolism Extension Package for ADaM in 'R' Asset Library

**Version** 0.1.0

**Description** A toolbox for programming Clinical Data Standards Interchange Consortium (CDISC) compliant Analysis Data Model (ADaM) datasets in R. ADaM datasets are a mandatory part of any New Drug or Biologics License Application submitted to the United States Food and Drug Administration (FDA). Analysis derivations are implemented in accordance with the ``Analysis Data Model Implementation Guide'' (CDISC Analysis Data Model Team, 2021, <https://www.cdisc.org/standards/foundational/adam>). The package is an extension package of the 'admiral' package focusing on the metabolism therapeutic area.

**License** Apache License (>= 2)

**URL** https://pharmaverse.github.io/admiralmetabolic/, https://github.com/pharmaverse/admiralmetabolic

**BugReports** https://github.com/pharmaverse/admiralmetabolic/issues

**Depends** R (>= 4.1)

**Imports** admiral (>= 1.1.1), admiraldev (>= 1.0.0), cli (>= 3.6.2), dplyr (>= 0.8.4), stringr (>= 1.4.0), lifecycle (>= 0.1.0), lubridate (>= 1.7.4), magrittr (>= 1.5), purrr (>= 0.3.3), rlang (>= 0.4.4), tidyselect (>= 1.0.0)

**Suggests** diffdf, DT, htmltools, knitr, methods, pharmaversesdtm (>= 1.0.0), rmarkdown, testthat (>= 3.0.0), tibble, usethis (>= 2.2.3)

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**Encoding** UTF-8

**Language** en-US

**LazyData** true

**RoxygenNote** 7.3.2

**NeedsCompilation** no

**Author** Anders Askeland [aut, cre] (<`https://orcid.org/0000-0002-2996-129X`>),
Andrii Yurovskyi [aut] (<`https://orcid.org/0009-0006-8606-0929`>),
Kathrin Flunkert [aut],
Edoardo Mancini [aut] (<`https://orcid.org/0009-0006-4899-8641`>),
Shunsuke Goto [aut],
Siddhesh Pujari [aut] (<`https://orcid.org/0009-0003-4971-1217`>),
Sonali Das [aut],
Olga Starostecka [aut],
Vang Le-Quy [aut] (<`https://orcid.org/0000-0003-2035-8619`>)

**Maintainer** Anders Askeland <`iakd@novonordisk.com`>

**Repository** CRAN

**Date/Publication** 2025-01-20 16:20:09 UTC

## Contents

---

## `derive_param_waisthgt` *Adds a Parameter for Waist to Height Ratio*

---

### Description

Adds a parameter for Waist to Height Ratio using Waist Circumference and Height for each by group (e.g., subject and visit) where the source parameters are available.

**Note:** This is a wrapper function for the more generic `admiral::derive_param_computed()`.

### Usage

```
derive_param_waisthgt(
  dataset,
  by_vars,
  wstcir_code = "WSTCIR",
  height_code = "HEIGHT",
  set_values_to = exprs(PARAMCD = "WAISTHGT"),
  filter = NULL,
  constant_by_vars = NULL,
  get_unit_expr
)
```

## Arguments

| | |
|---|---|
| `dataset` | Input dataset<br>The variables specified by the `by_vars` argument are expected to be in the dataset. `PARAMCD`, and `AVAL` are expected as well.<br>The variable specified by `by_vars` and `PARAMCD` must be a unique key of the input dataset after restricting it by the filter condition (`filter` argument) and to the parameters specified by `wstcir_code` and `height_code`. |
| `by_vars` | Grouping variables<br>For each group defined by `by_vars` an observation is added to the output dataset. Only variables specified in `by_vars` will be populated in the newly created records.<br>*Permitted Values*: list of variables created by `exprs()` e.g. `exprs(USUBJID, VISIT)` |
| `wstcir_code` | Waist Circumference parameter code<br>The observations where `PARAMCD` equals the specified value are considered as the Waist Circumference.<br>*Permitted Values:* character value |
| `height_code` | Height parameter code<br>The observations where `PARAMCD` equals the specified value are considered as the Height.<br>*Permitted Values:* character value |
| `set_values_to` | Variables to be set<br>The specified variables are set to the specified values for the new observations. For example `exprs(PARAMCD = "RATIO")` defines the parameter code for the new parameter.<br>*Permitted Values:* List of variable-value pairs |
| `filter` | Filter condition<br>The specified condition is applied to the input dataset before deriving the new parameter, i.e., only observations fulfilling the condition are taken into account.<br>*Permitted Values:* a condition |
| `constant_by_vars` | By variables for when Height is constant<br>When Height is constant, the Height parameters (measured only once) are merged to the other parameters using the specified variables.<br>If Height is constant (e.g. only measured once at screening or baseline) then use `constant_by_vars` to select the subject-level variable to merge on (e.g. `USUBJID`). This will produce Waist to Height Ratio at all visits where Waist Circumference is measured. Otherwise it will only be calculated at visits with both Height and Waist Circumference collected.<br>*Permitted Values*: list of variables created by `exprs()`, e.g. `exprs(USUBJID, VISIT)` |
| `get_unit_expr` | An expression providing the unit of the parameter<br>The result is used to check the units of the input parameters. If the units are not consistent within each parameter, an error will be thrown. |

Additionally, if the input parameters are measured in different units but are mutually convertible (e.g., centimeters for one parameter and inches for another), an automatic conversion will be performed in order to uniform the values before calculating the ratio.

**Note:** Conversion factors come from unit definitions as per CDISC standards. *m* is defined as 100 cm *mm* is defined as 0.1 cm *in* is defined as 2.54 cm *ft* is defined as 30.48 cm

*Permitted Values:* A variable of the input dataset or a function call

## Details

The analysis value of the new parameter is derived as

$$WAISTHGT = \frac{WSTCIR}{HEIGHT}$$

## Value

The input dataset with the new parameter added. Note, a variable will only be populated in the new parameter rows if it is specified in `by_vars`.

## See Also

`admiral::derive_param_computed()`

ADVS Functions for adding Parameters: `derive_param_waisthip()`

## Examples

```
library(tibble)
library(rlang)

# Example 1: Derive Waist to Height Ratio where Height is measured only once

advs <- tribble(
  ~USUBJID, ~PARAMCD, ~PARAM, ~AVAL, ~AVALU, ~VISIT,
  "01-101-1001", "HEIGHT", "Height (cm)", 147, "cm", "SCREENING",
  "01-101-1001", "WSTCIR", "Waist Circumference (cm)", 110, "cm", "SCREENING",
  "01-101-1001", "WSTCIR", "Waist Circumference (cm)", 108, "cm", "WEEK 2",
  "01-101-1001", "WSTCIR", "Waist Circumference (cm)", 107, "cm", "WEEK 3",
  "01-101-1002", "HEIGHT", "Height (cm)", 163, "cm", "SCREENING",
  "01-101-1002", "WSTCIR", "Waist Circumference (cm)", 120, "cm", "SCREENING",
  "01-101-1002", "WSTCIR", "Waist Circumference (cm)", 118, "cm", "WEEK 2",
  "01-101-1002", "WSTCIR", "Waist Circumference (cm)", 117, "cm", "WEEK 3",
)

derive_param_waisthgt(
  advs,
  by_vars = exprs(USUBJID, VISIT),
  wstcir_code = "WSTCIR",
  height_code = "HEIGHT",
  set_values_to = exprs(
    PARAMCD = "WAISTHGT",
    PARAM = "Waist to Height Ratio"
  ),
  constant_by_vars = exprs(USUBJID),
  get_unit_expr = admiral::extract_unit(PARAM)
)

# Example 2: Same as above but only adding Waist to Height Ratio
# at certain visits

derive_param_waisthgt(
  advs,
  by_vars = exprs(USUBJID, VISIT),
  wstcir_code = "WSTCIR",
  height_code = "HEIGHT",
  set_values_to = exprs(
    PARAMCD = "WAISTHGT",
    PARAM = "Waist to Height Ratio"
  ),
  constant_by_vars = exprs(USUBJID),
  get_unit_expr = admiral::extract_unit(PARAM),
  filter = VISIT %in% c("SCREENING", "WEEK 3")
)

# Example 3: Pediatric study where Height and Waist Circumference
# are measured multiple times

advs <- tribble(
  ~USUBJID, ~PARAMCD, ~PARAM, ~AVAL, ~AVALU, ~VISIT,
  "01-101-1001", "HEIGHT", "Height (cm)", 147, "cm", "SCREENING",
  "01-101-1001", "HEIGHT", "Height (cm)", 148, "cm", "WEEK 2",
  "01-101-1001", "HEIGHT", "Height (cm)", 149, "cm", "WEEK 3",
  "01-101-1001", "WSTCIR", "Waist Circumference (cm)", 100, "cm", "SCREENING",
  "01-101-1001", "WSTCIR", "Waist Circumference (cm)", 99, "cm", "WEEK 2",
  "01-101-1001", "WSTCIR", "Waist Circumference (cm)", 98, "cm", "WEEK 3",
  "01-101-1002", "HEIGHT", "Height (cm)", 163, "cm", "SCREENING",
  "01-101-1002", "HEIGHT", "Height (cm)", 164, "cm", "WEEK 2",
  "01-101-1002", "HEIGHT", "Height (cm)", 165, "cm", "WEEK 3",
  "01-101-1002", "WSTCIR", "Waist Circumference (cm)", 110, "cm", "SCREENING",
  "01-101-1002", "WSTCIR", "Waist Circumference (cm)", 109, "cm", "WEEK 2",
  "01-101-1002", "WSTCIR", "Waist Circumference (cm)", 108, "cm", "WEEK 3"
)

derive_param_waisthgt(
  advs,
  by_vars = exprs(USUBJID, VISIT),
  wstcir_code = "WSTCIR",
  height_code = "HEIGHT",
  set_values_to = exprs(
    PARAMCD = "WAISTHGT",
    PARAM = "Waist to Height Ratio"
  ),
  get_unit_expr = admiral::extract_unit(PARAM)
```

```
)

# Example 4: Automatic conversion is performed when deriving the ratio
# if parameters are provided in different units (e.g. centimeters and inches)

advs <- tribble(
  ~USUBJID, ~PARAMCD, ~PARAM, ~AVAL, ~AVALU, ~VISIT,
  "01-101-1001", "HEIGHT", "Height (cm)", 147, "cm", "SCREENING",
  "01-101-1001", "WSTCIR", "Waist Circumference (in)", 39.37, "in", "SCREENING",
  "01-101-1001", "WSTCIR", "Waist Circumference (in)", 38.98, "in", "WEEK 2",
  "01-101-1001", "WSTCIR", "Waist Circumference (in)", 38.58, "in", "WEEK 3",
  "01-101-1002", "HEIGHT", "Height (cm)", 163, "cm", "SCREENING",
  "01-101-1002", "WSTCIR", "Waist Circumference (in)", 43.31, "in", "SCREENING",
  "01-101-1002", "WSTCIR", "Waist Circumference (in)", 42.91, "in", "WEEK 2",
  "01-101-1002", "WSTCIR", "Waist Circumference (in)", 42.52, "in", "WEEK 3"
)

derive_param_waisthgt(
  advs,
  by_vars = exprs(USUBJID, VISIT),
  wstcir_code = "WSTCIR",
  height_code = "HEIGHT",
  set_values_to = exprs(
    PARAMCD = "WAISTHGT",
    PARAM = "Waist to Height Ratio"
  ),
  constant_by_vars = exprs(USUBJID),
  get_unit_expr = admiral::extract_unit(PARAM)
)
```

## derive_param_waisthip *Adds a Parameter for Waist to Hip Ratio*

### Description

Adds a parameter for Waist to Hip Ratio using Waist Circumference and Hip Circumference for each by group (e.g., subject and visit) where the source parameters are available.

**Note:** This is a wrapper function for the more generic `admiral::derive_param_computed()`.

### Usage

```
derive_param_waisthip(
  dataset,
  by_vars,
  wstcir_code = "WSTCIR",
  hipcir_code = "HIPCIR",
  set_values_to = exprs(PARAMCD = "WAISTHIP"),
  filter = NULL,
  get_unit_expr
)
```

## Arguments

| | |
|---|---|
| `dataset` | Input dataset<br>The variables specified by the `by_vars` argument are expected to be in the dataset. `PARAMCD`, and `AVAL` are expected as well.<br>The variable specified by `by_vars` and `PARAMCD` must be a unique key of the input dataset after restricting it by the filter condition (`filter` argument) and to the parameters specified by `wstcir_code` and `hipcir_code`. |
| `by_vars` | Grouping variables<br>For each group defined by `by_vars` an observation is added to the output dataset. Only variables specified in `by_vars` will be populated in the newly created records.<br>*Permitted Values*: list of variables created by `exprs()` e.g. `exprs(USUBJID, VISIT)` |
| `wstcir_code` | Waist Circumference parameter code<br>The observations where `PARAMCD` equals the specified value are considered as the Waist Circumference.<br>*Permitted Values:* character value |
| `hipcir_code` | Hip Circumference parameter code<br>The observations where `PARAMCD` equals the specified value are considered as the Hip Circumference<br>*Permitted Values:* character value |
| `set_values_to` | Variables to be set<br>The specified variables are set to the specified values for the new observations. For example `exprs(PARAMCD = "RATIO")` defines the parameter code for the new parameter.<br>*Permitted Values:* List of variable-value pairs |
| `filter` | Filter condition<br>The specified condition is applied to the input dataset before deriving the new parameter, i.e., only observations fulfilling the condition are taken into account.<br>*Permitted Values:* a condition |
| `get_unit_expr` | An expression providing the unit of the parameter<br>The result is used to check the units of the input parameters. If the units are not consistent within each parameter, an error will be thrown.<br>Additionally, if the input parameters are measured in different units but are mutually convertible (e.g., centimeters for one parameter and inches for another), an automatic conversion will be performed in order to uniform the values before calculating the ratio.<br>**Note:** Conversion factors come from unit definitions as per CDISC standards. *m* is defined as 100 cm *mm* is defined as 0.1 cm *in* is defined as 2.54 cm *ft* is defined as 30.48 cm<br>*Permitted Values:* A variable of the input dataset or a function call |

## Details

The analysis value of the new parameter is derived as

$$WAISTHIP = \frac{WSTCIR}{HIPCIR}$$

## Value

The input dataset with the new parameter added. Note, a variable will only be populated in the new parameter rows if it is specified in `by_vars`.

## See Also

`admiral::derive_param_computed()`

ADVS Functions for adding Parameters: `derive_param_waisthgt()`

## Examples

```
library(tibble)
library(rlang)

advs <- tribble(
  ~USUBJID, ~PARAMCD, ~PARAM, ~AVAL, ~AVALU, ~VISIT,
  "01-101-1001", "WSTCIR", "Waist Circumference (cm)", 110, "cm", "SCREENING",
  "01-101-1001", "WSTCIR", "Waist Circumference (cm)", 108, "cm", "WEEK 2",
  "01-101-1001", "WSTCIR", "Waist Circumference (cm)", 107, "cm", "WEEK 3",
  "01-101-1001", "HIPCIR", "Hip Circumference (cm)", 125, "cm", "SCREENING",
  "01-101-1001", "HIPCIR", "Hip Circumference (cm)", 124, "cm", "WEEK 2",
  "01-101-1001", "HIPCIR", "Hip Circumference (cm)", 123, "cm", "WEEK 3",
  "01-101-1002", "WSTCIR", "Waist Circumference (cm)", 120, "cm", "SCREENING",
  "01-101-1002", "WSTCIR", "Waist Circumference (cm)", 118, "cm", "WEEK 2",
  "01-101-1002", "WSTCIR", "Waist Circumference (cm)", 117, "cm", "WEEK 3",
  "01-101-1002", "HIPCIR", "Hip Circumference (cm)", 135, "cm", "SCREENING",
  "01-101-1002", "HIPCIR", "Hip Circumference (cm)", 133, "cm", "WEEK 2",
  "01-101-1002", "HIPCIR", "Hip Circumference (cm)", 132, "cm", "WEEK 3"
)

derive_param_waisthip(
  advs,
  by_vars = exprs(USUBJID, VISIT),
  wstcir_code = "WSTCIR",
  hipcir_code = "HIPCIR",
  set_values_to = exprs(
    PARAMCD = "WAISTHIP",
    PARAM = "Waist to Hip Ratio"
  ),
  get_unit_expr = admiral::extract_unit(PARAM)
)

# Only adding Waist to Hip Ratio at certain visits

derive_param_waisthip(
  advs,
  by_vars = exprs(USUBJID, VISIT),
  wstcir_code = "WSTCIR",
  hipcir_code = "HIPCIR",
  set_values_to = exprs(
    PARAMCD = "WAISTHIP",
    PARAM = "Waist to Hip Ratio"
  ),
  get_unit_expr = admiral::extract_unit(PARAM),
  filter = VISIT %in% c("SCREENING", "WEEK 3")
)

# Automatic conversion is performed when deriving the ratio
# if parameters are provided in different units

advs <- tribble(
  ~USUBJID, ~PARAMCD, ~PARAM, ~AVAL, ~AVALU, ~VISIT,
  "01-101-1001", "HIPCIR", "Hip Circumference (cm)", 125, "cm", "SCREENING",
  "01-101-1001", "HIPCIR", "Hip Circumference (cm)", 124, "cm", "WEEK 2",
  "01-101-1001", "HIPCIR", "Hip Circumference (cm)", 123, "cm", "WEEK 3",
  "01-101-1001", "WSTCIR", "Waist Circumference (in)", 43.31, "in", "SCREENING",
  "01-101-1001", "WSTCIR", "Waist Circumference (in)", 42.52, "in", "WEEK 2",
  "01-101-1001", "WSTCIR", "Waist Circumference (in)", 42.13, "in", "WEEK 3",
  "01-101-1002", "HIPCIR", "Hip Circumference (cm)", 135, "cm", "SCREENING",
  "01-101-1002", "HIPCIR", "Hip Circumference (cm)", 133, "cm", "WEEK 2",
  "01-101-1002", "HIPCIR", "Hip Circumference (cm)", 132, "cm", "WEEK 3",
  "01-101-1002", "WSTCIR", "Waist Circumference (in)", 47.24, "in", "SCREENING",
  "01-101-1002", "WSTCIR", "Waist Circumference (in)", 46.46, "in", "WEEK 2",
  "01-101-1002", "WSTCIR", "Waist Circumference (in)", 46.06, "in", "WEEK 3"
)

derive_param_waisthip(
  advs,
  by_vars = exprs(USUBJID, VISIT),
  wstcir_code = "WSTCIR",
  hipcir_code = "HIPCIR",
  set_values_to = exprs(
    PARAMCD = "WAISTHIP",
    PARAM = "Waist to Hip Ratio"
  ),
  get_unit_expr = admiral::extract_unit(PARAM)
)
```

## dm_metabolic *Example demographic dataset*

### Description

An example Demographic SDTM dataset for metabolic studies.

### Usage

```
dm_metabolic
```

### Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 5 rows and 25 columns.

### Source

Constructed using `dm` from the `{pharmaversesdtm}` package

---

## `qs_metabolic` *Example questionnaire dataset*

---

### Description

An example questionnaire SDTM dataset (containing Control of Eating Questionnaire, CoEQ, test data) for metabolic studies. Note that University of Leeds are the copyright holders of the CoEQ and the test data included within `{admiralmetabolic}` is for not-for-profit use only within `{admiralmetabolic}` and `pharmaverse`-related examples/documentation. Any persons or companies wanting to use the CoEQ should request a license to do so from the following link.

### Usage

```
qs_metabolic
```

### Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 966 rows and 18 columns.

### Source

Constructed using `dm_metabolic` from `{admiralmetabolic}` package

---

## `vs_metabolic` *Example vital signs dataset*

---

### Description

An example Vital Signs SDTM dataset for metabolic studies.

### Usage

```
vs_metabolic
```

## Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 719 rows and 24 columns.

## Source

Constructed using `vs` from the `{pharmaversesdtm}` package and `dm_metabolic` from `{admiralmetabolic}` package

# Index