

Package: ckbtools (via r-universe)

October 11, 2026

Title Tools for Analyses in Epidemiology

Version 0.1.0

Description A selection of tools for analyses in epidemiology.

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

Suggests asht, knitr, rmarkdown, withr, testthat (>= 3.3.0)

VignetteBuilder knitr

URL <https://ckbiobank.github.io/ckbtools>

Imports broom, cli, dplyr, emmeans, Epi, glue, lubridate, magrittr,
purrr, qvcalc, rlang, survival, tibble, tidyr, tidyselect

LazyData true

Depends R (>= 3.5)

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

Config/pak/sysreqs libicu-dev

Repository <https://ckbiobank.r-universe.dev>

Date/Publication 2026-08-25 10:19:57 UTC

RemoteUrl <https://github.com/ckbiobank/ckbtools>

RemoteRef HEAD

RemoteSha 8aac9a67f8dfa092f1bdd3785f0106f03a8d0efc

Contents

ckbtools_ascertainments	2
ckbtools_baseline	3
ckbtools_endpoint_definitions	3
ckbtools_endpoints	3
ckbtools_participant_data	4

ckbtools_raw_olink_meta_data	4
ckbtools_raw_olink_protein_data	4
ckbtools_raw_somascan_meta_data	5
ckbtools_raw_somascan_protein_data	5
ckbtools_raw_somascan_sample_data	5
cox_model	6
expand_age_at_risk	8
heterogeneity	9
incidence_rates	10
linear_model	11
logistic_model	13
multiple_models	15
prepare_olink_data	18
prepare_olink_meta_data	19
prepare_proteomics_analysis_dataset	20
prepare_somascan_data	21
prepare_somascan_meta_data	22
safemodels	23
summarise_continuous	23
trend	24
Index	25

ckbtools_ascertainments
<i>Synthetic CKB ascertainment data for demoing functions</i>

Description

Synthetic CKB ascertainment data for demoing functions

Usage

ckbtools_ascertainments

Format

An object of class data.frame with 25000 rows and 3 columns.

ckbtools_baseline	<i>Synthetic CKB baseline data for demoing functions</i>
-------------------	--

Description

Synthetic CKB baseline data for demoing functions

Usage

ckbtools_baseline

Format

An object of class data.frame with 25000 rows and 26 columns.

ckbtools_endpoint_definitions	<i>Synthetic CKB endpoint definitions data for demoing functions</i>
-------------------------------	--

Description

Synthetic CKB endpoint definitions data for demoing functions

Usage

ckbtools_endpoint_definitions

Format

An object of class data.frame with 2 rows and 2 columns.

ckbtools_endpoints	<i>Synthetic CKB endpoint data for demoing functions</i>
--------------------	--

Description

Synthetic CKB endpoint data for demoing functions

Usage

ckbtools_endpoints

Format

An object of class data.frame with 25000 rows and 5 columns.

ckbtools_participant_data

Synthetic CKB participant data for demoing functions

Description

Synthetic CKB participant data for demoing functions

Usage

ckbtools_participant_data

Format

An object of class `data.frame` with 25000 rows and 47 columns.

ckbtools_raw_olink_meta_data

Synthetic CKB Olink proteomics data for demoing functions

Description

Synthetic CKB Olink proteomics data for demoing functions

Usage

ckbtools_raw_olink_meta_data

Format

An object of class `data.frame` with 48 rows and 3 columns.

ckbtools_raw_olink_protein_data

Synthetic CKB Olink proteomics data for demoing functions

Description

Synthetic CKB Olink proteomics data for demoing functions

Usage

ckbtools_raw_olink_protein_data

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 48000 rows and 11 columns.

`ckbtools_raw_somascan_meta_data`*Synthetic CKB SomaScan proteomics data for demoing functions*

Description

Synthetic CKB SomaScan proteomics data for demoing functions

Usage`ckbtools_raw_somascan_meta_data`**Format**

An object of class `data.frame` with 48 rows and 7 columns.

`ckbtools_raw_somascan_protein_data`*Synthetic CKB SomaScan proteomics data for demoing functions*

Description

Synthetic CKB SomaScan proteomics data for demoing functions

Usage`ckbtools_raw_somascan_protein_data`**Format**

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 48000 rows and 4 columns.

`ckbtools_raw_somascan_sample_data`*Synthetic CKB SomaScan proteomics data for demoing functions*

Description

Synthetic CKB SomaScan proteomics data for demoing functions

Usage`ckbtools_raw_somascan_sample_data`**Format**

An object of class `data.frame` with 1000 rows and 5 columns.

 cox_model

Estimate the association between an exposure and incident disease

Description

This function is used to estimate the (adjusted) association between an exposure variable and an incident disease endpoint using a Cox proportional hazards model. It is a wrapper for the [coxph](#) function for convenience and provides results from some additional analysis steps if requested.

Usage

```
cox_model(
  data,
  exposure,
  outcome,
  outcome_date = paste0(outcome, "_date"),
  start_date = "study_date",
  time_in = NULL,
  time_out = NULL,
  adjust_for = NULL,
  exclude = NULL,
  exposure_term = exposure,
  cluster = NULL,
  init_values = c(0),
  additional = "",
  return = c("model", "reduced-model", "coefs", "exposure"),
  return_fn = identity,
  ...
)
```

Arguments

<code>data</code>	A data frame.
<code>exposure</code>	Name of column containing exposure variable.
<code>outcome</code>	Name of the column containing status indicator (normally 0 = no disease, 1 = disease).
<code>outcome_date</code>	Name of column containing date of event or censoring date. (Default: <code>paste0(outcome, "_date")</code>)
<code>start_date</code>	Name of column containing date of start of follow up. (Default: <code>"study_date"</code>)
<code>time_in</code>	Name of column containing starting time for interval data.
<code>time_out</code>	Name of column containing follow up time (or ending time for interval data).
<code>adjust_for</code>	Character vector of terms to include in the model formula.
<code>exclude</code>	Names of columns that identify observations that should be excluded.
<code>exposure_term</code>	The term(s) to include in the model for the exposure. This can be a character vector or a function that take the exposure name as the argument and returns a character vector. (Default: <code>exposure</code>)

cluster	Name of column that clusters the observations, for the purposes of a robust variance.
init_values	A named vector of initial values for model coefficients.
additional	Extra results to include. See "Additional results" for details.
return	What should the function return: • "model" (default): The model object (see ?survival::coxph.object). • "reduced-model": The model object with some large components removed. • "coefs": A data frame of all estimated coefficients and other key model statistics. • "exposure": A data frame of estimated coefficient(s) for the exposure and other key model statistics.
return_fn	If return="model", name of function to apply to model before it's returned. (Default: identity)
...	Arguments passed to survival::coxph()

Value

Either a model object, reduced model object or data frame of coefficients.

Time scale

For analysis of incident disease using time-on-study as the time scale, outcome, outcome_date and start_date are used to define follow up time (right censored data).

For other analyses, the time_in and time_out arguments can be used. If time_in and time_out are given, then these are starting and ending time for interval survival data. If time_out (but not time_in) is given, then this is the follow up time (right censored data).

Additional results

The additional argument is used to add results from additional analyses. It must be a character vector containing one or more of:

- "lrt": Likelihood ratio test comparing the full model to one without the exposure. Output will have additional columns "Chisq" (chi-square test statistics), "Df" (degrees of freedom), "p_Chisq" (p-value).
- "float": Standard errors for the exposure calculated by [float](#). Output will have additional columns "float_estimate", "float_se" and "limits".
- "qvcalc": Standard errors for the exposure calculated by [qvcalc](#). Output will have additional columns "qvcalc_estimate", "quasi_se" and "worstErrors".
- "" (default): Nothing.

These will only be included when relevant e.g. standard errors from [float](#) when the exposure has more than two levels. This only has an effect if return is "coefs" or "exposure".

See [float](#) and [qvcalc](#) for details about these results. Note that "limits" and "worstErrors" do not correspond to particular groups but accuracy over all contrasts.

Examples

```
cox_model(ckbtools_participant_data,
  exposure = "bmi_calc",
  outcome = "all_cause_mortality",
  adjust_for = c("age_at_study_date", "strata(sex)"),
  return = "coefs")
```

```
cox_model(ckbtools_participant_data,
  exposure = "bmi_grp",
  outcome = "all_cause_mortality",
  adjust_for = c("age_at_study_date", "strata(sex)"),
  return = "coefs")
```

expand_age_at_risk	<i>Lexis expansion of data by age-at-risk</i>
--------------------	---

Description

Lexis expansion of data by age-at-risk

Usage

```
expand_age_at_risk(
  data,
  col.id = "csid",
  col.dob = "dob_anon",
  col.entry_date = "study_date",
  col.exit_date = "endpoint_date",
  col.status = "endpoint",
  col.keep = NULL,
  ages = c(0, 100),
  age.labels = paste0(ages[-length(ages)], "-", ages[-1]),
  null.status = 0
)
```

Arguments

data	A data frame
col.id	Name of column containing unique participant IDs
col.dob	Name of column containing date of birth (Must be of class Date)
col.entry_date	Name of column containing entry date (Must be of class Date)
col.exit_date	Name of column containing exit date (Must be of class Date)
col.status	Name of column containing status
col.keep	Names of additional columns to include in the result (e.g. exposure and covariate variables)

<code>ages</code>	Vector of ages (years) to create age-at-risk strata. Any participant follow-up outside these ages is dropped. (Must be a numeric vector with distinct, increasing values. Default: <code>c(0, 100)</code>)
<code>age.labels</code>	Vector of name for age-at-risk strata (Default: <code>paste0(ages[-length(ages)], "-", ages[-1])</code>)
<code>null.status</code>	Value to use in status column for intervals before the final interval (Default: 0)

Value

Data frame expanded by age-at-risk

heterogeneity	<i>Test for heterogeneity</i>
---------------	-------------------------------

Description

A chi-squared test (on $k - 1$ degrees of freedom) for heterogeneity given a vector of k independent estimates and their standard errors.

Usage

```
heterogeneity(beta, se)
```

Arguments

<code>beta</code>	vector of independent estimates
<code>se</code>	vector of their standard errors

Details

See `vignette("trend_heterogeneity")` for details.

Value

A named list:

- test statistic (chi-squared test statistic)
- degrees of freedom (number of degrees of freedom used for the test, equal to `length(beta) - 1`)
- p (p-value)

incidence_rates	<i>Calculate standardised incidence rates</i>
-----------------	---

Description

Calculate standardised incidence rates

Usage

```
incidence_rates(
  data,
  col.dob = "dob_anon",
  col.entry_date = "study_date",
  col.exit_date = "endpoint_date",
  col.status = "endpoint",
  ages = c(0, 100),
  group = NULL,
  adjust_for = NULL,
  per_pyar = 1000,
  weights = NULL,
  method = c("norm", "gamma")
)
```

Arguments

data	A data frame.
col.dob	Name of column containing date of birth. (Must be of class Date.)
col.entry_date	Name of column containing entry date. (Must be of class Date.)
col.exit_date	Name of column containing exit date. (Must be of class Date.)
col.status	Name of column containing status. This should be 1 if the event has occurred at the exit date, or 0 if it has not.
ages	Vector of ages (years) to create age-at-risk strata. (Must be a numeric vector with distinct, increasing values. Default: c(0, 100))
group	Name of column containing groups to separately calculate incidence rates.
adjust_for	Vector of names of columns containing adjustment variables. Include "agegrp" to adjust for age-at-risk strata.
per_pyar	Crude and standardised incidence rates are given per per_pyar person-years at risk. (Default: 1000)
weights	A data frame with columns that match the adjust_for variables, and one row for each combination of levels of those variables, and a column named 'std_pyar' with the weights (person-years at risk for that stratum in the reference/standard population).
method	Method to calculate confidence intervals, one of "norm" or "gamma". Default "norm" is Normal approximation.

Value

A named list:

- incidence_rates_in_all_strata
- standardised_incidence_rates

Methods

See vignette("incidence_rates") for details.

linear_model	<i>Estimate the association between an exposure and a continuous outcome</i>
--------------	--

Description

This function is used to estimate the (adjusted) linear association between an exposure variable and a continuous outcome variable using a linear regression model. It is a wrapper for the [lm](#) function for convenience and provides results from some additional analysis steps if requested.

Usage

```
linear_model(
  data,
  exposure,
  outcome,
  adjust_for = NULL,
  exclude = NULL,
  exposure_term = exposure,
  additional = "",
  return = c("model", "reduced-model", "coefs", "exposure"),
  return_fn = identity,
  nuisance = character(0),
  ...
)
```

Arguments

data	A data frame.
exposure	Name of column containing exposure variable.
outcome	Name of column containing outcome variable.
adjust_for	Character vector of terms to include in the model formula.
exclude	Names of columns that identify observations that should be excluded.
exposure_term	The term(s) to include in the model for the exposure. This can be a character vector or a function that take the exposure name as the argument and returns a character vector. (Default: exposure)

additional	Extra results to include. See "Additional results" for details.
return	What should the function return: • "model" (default): The model object (object of class "lm"). • "reduced-model": The model object with some large components removed. • "coefs": A data frame of all estimated coefficients and other key model statistics. • "exposure": A data frame of estimated coefficient(s) for the exposure and other key model statistics.
return_fn	If return="model", name of function to apply to model before it's returned. (Default: identity)
nuisance	If "emmeans" is requested via the additional argument, use this to specify predictors to omit from the reference grid. See the help for emmeans and ref_grid .
...	Arguments passed to <code>stats::lm()</code>

Value

Either a model object, reduced model object or data frame of coefficients.

Additional results

The additional argument is used to add results from additional analyses. It must be a character vector containing one or more of:

- "anova": ANOVA F-test comparing the full model to one without the exposure. Output will have additional columns "F" (F test statistics), "Df" (degrees of freedom), "p_F" (p-value).
- "float": Standard errors for the exposure calculated by [float](#). Output will have additional columns "float_estimate", "float_se" and "limits".
- "qvcalc": Standard errors for the exposure calculated by [qvcalc](#). Output will have additional columns "qvcalc_estimate", "quasi_se" and "worstErrors".
- "emmeans": Estimated marginal means for the exposure calculated using [emmeans](#) with proportional weights for factor variables that are averaged over. Output will have additional columns "emmean", "emmean_se" and "emmean_quasi_se".
- "" (default): Nothing.

These will only be included when relevant e.g. standard errors from [float](#) when the exposure has more than two levels. This only has an effect if return is "coefs" or "exposure".

See [float](#) and [qvcalc](#) for details about these results. Note that "limits" and "worstErrors" do not correspond to particular groups but accuracy over all contrasts.

Examples

```
linear_model(ckbtools_participant_data,
             exposure = "age_at_study_date",
             outcome = "sbp_mean",
             adjust_for = "sex",
             return = "coefs")
```

```
# factor exposure, calculate estimated marginal means
linear_model(ckbtools_participant_data,
             exposure = "smoking",
             outcome = "sbp_mean",
             adjust_for = c("age_at_study_date", "sex"),
             additional = "emmeans",
             return = "coefs")

# return the model object
linear_model(ckbtools_participant_data,
             exposure = "bmi_grp",
             outcome = "sbp_mean",
             adjust_for = c("age_at_study_date", "sex"),
             return = "model")
```

logistic_model

Estimate the association between an exposure and a binary outcome

Description

This function is used to estimate the (adjusted) association (log odds ratio) between an exposure variable and a binary outcome variable using a logistic regression model. It is a wrapper for the [glm](#) function with family = "binomial" for convenience and provides results from some additional analysis steps if requested.

Usage

```
logistic_model(
  data,
  exposure,
  outcome,
  adjust_for = NULL,
  exclude = NULL,
  exclude_controls = NULL,
  exposure_term = exposure,
  additional = "",
  return = c("model", "reduced-model", "coefs", "exposure"),
  return_fn = identity,
  ...
)
```

Arguments

data	A data frame.
exposure	Name of column containing exposure variable.
outcome	Name of the column containing binary outcome variable.
adjust_for	Character vector of terms to include in the model formula.

<code>exclude</code>	Names of columns that identify observations that are to be excluded.
<code>exclude_controls</code>	Names of columns that identify observations that are to be excluded when outcome is equal to zero.
<code>exposure_term</code>	The term(s) to include in the model for the exposure. This can be a character vector or a function that take the exposure name as the argument and returns a character vector. (Default: <code>exposure</code>)
<code>additional</code>	Extra results to include. See "Additional results" for details.
<code>return</code>	What should the function return: • <code>"model"</code> (default): The model object (object of class <code>"glm"</code>). • <code>"reduced-model"</code>: The model object with some large components removed. • <code>"coefs"</code>: A data frame of all estimated coefficients and other key model statistics. • <code>"exposure"</code>: A data frame of estimated coefficient(s) for the exposure and other key model statistics.
<code>return_fn</code>	If <code>return="model"</code> , name of function to apply to model before it's returned. (Default: <code>identity</code>)
<code>...</code>	Arguments passed to <code>stats::glm()</code>

Value

Either a model object, reduced model object or data frame of coefficients.

Additional results

The `additional` argument is used to add results from additional analyses. It must be a character vector containing one or more of:

- `"lrt"`: Likelihood ratio test comparing the full model to one without the exposure. Output will have additional columns `"Chisq"` (chi-square test statistics), `"Df"` (degrees of freedom), `"p_Chisq"` (p-value).
- `"float"`: Standard errors for the exposure calculated by [float](#) or by [ftrend](#) if the model has no intercept term. Output will have additional columns `"float_estimate"`, `"float_se"` and `"limits"`.
- `"qvcalc"`: Standard errors for the exposure calculated by [qvcalc](#). Output will have additional columns `"qvcalc_estimate"`, `"quasi_se"` and `"worstErrors"`.
- `""` (default): Nothing.

These will only be included when relevant e.g. standard errors from [float](#) when the exposure has more than two levels. This only has an effect if `return` is `"coefs"` or `"exposure"`.

See [float](#) and [qvcalc](#) for details about these results. Note that `"limits"` and `"worstErrors"` do not correspond to particular groups but accuracy over all contrasts.

Examples

```
logistic_model(ckbtools_participant_data,
               exposure = "bmi_calc",
               outcome = "has_diabetes",
               adjust_for = c("age_at_study_date", "sex"),
               return = "coefs")

logistic_model(ckbtools_participant_data,
               exposure = "bmi_grp",
               outcome = "has_diabetes",
               adjust_for = c("age_at_study_date", "sex"),
               return = "coefs")
```

multiple_models	<i>Multiple linear, logistic or Cox proportional hazards regression models</i>
-----------------	--

Description

Multiple linear, logistic or Cox proportional hazards regression models

Usage

```
multiple_models(
  data,
  model,
  exposures,
  outcomes,
  outcome_dates = paste0(outcomes, "_date"),
  start_date = "study_date",
  time_in = NULL,
  time_out = NULL,
  age_at_risk = NULL,
  by = NULL,
  exclude = NULL,
  exclude_controls = NULL,
  exclude_by_exposure = NULL,
  exclude_by_outcome = NULL,
  exclude_controls_by_outcome = NULL,
  adjust_for = list(none = NULL),
  remove_adjust_for = NULL,
  add_adjust_for = NULL,
  cluster = NULL,
  init_values = NULL,
  return = c("coefs", "exposure", "reduced-model", "model"),
  verbose = if (rlang::is_interactive()) "progress" else "updates",
  dryrun = FALSE,
```

```
    ...  
  )
```

Arguments

<code>data</code>	A data frame.
<code>model</code>	Type of model to use: • "linear": Linear regression using <code>linear_model_safely()</code>. • "logistic": Logistic regression using <code>logistic_model_safely()</code>. • "cox": Cox proportional hazards model using <code>cox_model_safely()</code>.
<code>exposures</code>	Names of columns containing exposure variables.
<code>outcomes</code>	Names of columns containing outcome variables.
<code>outcome_dates</code>	Names of columns containing outcome dates (for Cox model only).
<code>start_date</code>	Name of column containing date of start of follow up. (Default: "study_date")
<code>time_in</code>	Names of columns containing starting time for interval data (for Cox model only).
<code>time_out</code>	Names of columns containing follow up time (or ending time for interval data) (for Cox model only).
<code>age_at_risk</code>	A list of arguments for <code>age_at_risk()</code> to expand data by age-at-risk for each outcome before fitting models for that outcome. If this is used, "strata(agegrp)" is automatically added to each element of <code>adjust_for</code> .
<code>by</code>	Name of column (or multiple columns) to create subgroups.
<code>exclude</code>	Names of columns that identify observations that should be excluded.
<code>exclude_controls</code>	Names of columns that identify observations that are to be excluded when outcome is equal to zero.
<code>exclude_by_exposure</code>	A named list of character vectors of column names. When the name of an element matches the exposure, the column names will be added to exclude.
<code>exclude_by_outcome</code>	A named list of character vectors of column names. When the name of an element matches the outcome, the column names will be added to exclude.
<code>exclude_controls_by_outcome</code>	A named list of character vectors of column names. When the name of an element matches the outcome, the column names will be added to <code>exclude_controls</code> .
<code>adjust_for</code>	A named list of character vectors of terms to include in model formulas.
<code>remove_adjust_for</code>	A named list of character vectors of terms. When the name of an element matches either the outcome or exposure, terms given in the character vector will be removed from every element of <code>adjust_for</code> .
<code>add_adjust_for</code>	A named list of character vectors of terms. When the name of an element matches either the outcome or exposure, terms given in the character vector will be added to every element of <code>adjust_for</code> .

cluster	Name of column that clusters the observations, for the purposes of a robust variance.
init_values	For Cox models, initial values for model coefficients. This must be a data frame with columns "outcome", "subgroup", "adjustment", "term" and "estimate", where term and estimate are the coefficient names and initial values.
return	What should the model function return: • "exposure" (default): A data frame of estimated coefficient(s) for the exposure and other key model statistics. • "coefs": A data frame of all estimated coefficients and other key model statistics. • "model": The model object. • "reduced-model": The model object with some large components removed. Note that R model objects can be very large (they can contain the model data, residuals, fitted values, etc.) so it is not recommended to set return="model" for a large number of models.
verbose	"progress" (default) to show progress bars, "updates" for messages without progress bars, or "quiet" to hide.
dryrun	Return only the data frame of models to be run. Columns (except 'subgroup', 'adjustment' and 'outcome_name') are the arguments that will be used in the model function. (Default: FALSE)
...	Arguments passed to the model function.

Value

A "tibble" data frame. If return is "model" or "reduced-model" then it will contain a list-column called "model".

Iteration

This function provides a convenient way to iterate over models using `linear_model()`, `logistic_model()` or `cox_model()`. First a data frame defining all models is created. If the number of exposures is greater than or equal to the number of outcomes (or Cox models with age-at-risk expansion are used) then the models are split by outcome, otherwise they are split by exposure. Within each set of models, `purrr::pmap()` is used to iterate over the fitting of models. Results are then combined into a single data frame.

For iteration of other models or analyses, you must write your own code. The Iteration chapter of the first edition of “R for Data Science” provides an introduction to iteration using R: <https://r4ds.had.co.nz/iteration.html>

See Also

`linear_model()`, `logistic_model()`, `cox_model()`, `linear_model_safely()`, `logistic_model_safely()`, `cox_model_safely()`

Examples

```
# linear regression models
multiple_models(
  ckbtools_participant_data,
  model = "linear",
  exposures = c("age_at_study_date", "sex", "bmi_calc", "bmi_grp"),
  outcomes = c("sbp_mean")
)

# logistic regression models
multiple_models(
  ckbtools_participant_data,
  model = "logistic",
  exposures = c("age_at_study_date", "sex", "bmi_calc", "bmi_grp"),
  outcomes = c("has_diabetes")
)

# Cox proportional hazards models
multiple_models(
  ckbtools_participant_data,
  model = "cox",
  exposures = c("age_at_study_date", "sex", "bmi_calc", "bmi_grp"),
  outcomes = c("all_cause_mortality", "ihd")
)
```

prepare_olink_data	<i>Preparation of Olink protein data</i>
--------------------	--

Description

Preparation of Olink protein data

Usage

```
prepare_olink_data(
  data,
  meta_data,
  transform = c("none"),
  qc = c("default", "none"),
  columns = NULL
)
```

Arguments

data	Raw Olink protein data (imported from data_baseline_olink_explore.rds or data_baseline_olink_explore.c)
meta_data	Olink meta data produced by prepare_olink_meta_data() .

transform	Transformation to apply to protein values. Currently only "none" is available.
qc	QC steps to apply: • "default" (default): replace protein values with a QC warning with a missing value • "none"
columns	Other columns in raw data to include in output. For example, set to "lod" to include the limit of detection for each protein.

```
prepare_olink_meta_data
```

Preparation of Olink meta data

Description

Preparation of Olink meta data

Usage

```
prepare_olink_meta_data(
  data,
  raw_olink_protein_data,
  names = c("protein", "id"),
  columns = c("uniprot", "olinkid")
)
```

Arguments

data	Raw CKB Olink meta data (imported from olink_explore_meta.rds or olink_explore_meta.csv)
raw_olink_protein_data	Raw Olink protein data (imported from data_baseline_olink_explore.rds or data_baseline_olink_explore.csv)
names	Naming scheme to use for proteins: "protein" (default) to use assay names or "id" to use Olink IDs.
columns	Optional columns to include in the meta data. (Default: c("uniprot", "olinkid"))

```
prepare_proteomics_analysis_dataset
```

Preparation of a proteomics analysis data set

Description

Preparation of a proteomics analysis data set

Usage

```
prepare_proteomics_analysis_dataset(
  participant_data,
  protein_data,
  protein_meta_data,
  plate_correction = c("none", "median-subcohort", "mean-subcohort"),
  scaling = c("none", "center-subcohort", "scale-subcohort", "center", "scale", "rint"),
  id = "csid"
)
```

Arguments

participant_data	A data frame of participant data
protein_data	Protein data produced by prepare_olink_data() or prepare_somascan_data() .
protein_meta_data	Corresponding protein meta data produced by prepare_olink_meta_data() or prepare_somascan_meta_data() .
plate_correction	Type of plate correction to apply: • "none" (default) • "median-subcohort": For each protein, subtract the plate median (of subcohort values) and add the overall median (of subcohort values). • "mean-subcohort": For each protein, subtract the plate mean (of subcohort values) and add the overall mean (of subcohort values).
scaling	Scaling to apply to values for each protein: • "none" (default) • "center-subcohort": Subtract the overall mean of subcohort values. • "scale-subcohort": Subtract the overall mean and divide by the sample standard deviation of subcohort values. • "center": Subtract the overall mean. • "scale": Subtract the overall mean and divide by the sample standard deviation. • "rint": Rank inverse normal transformation.
id	Name of column used to join participant_data and protein_data. (Default: "csid")

```
prepare_somascan_data Prepare SomaScan protein data
```

Description

This functions prepares a data set of SomaScan protein data given the raw CKB SomaScan data.

Usage

```
prepare_somascan_data(
  data,
  sample_data,
  meta_data,
  transform = c("log2", "log", "log10", "none"),
  qc = c("default", "flagged", "none"),
  repeat_samples = c("remove", "retain"),
  columns = NULL
)
```

Arguments

data	Raw CKB SomaScan protein data (imported from data_baseline_somalologic.rds or data_baseline_somalologic.csv)
sample_data	Raw CKB SomaScan protein sample information data (imported from data_baseline_somalologic_samples.rds or data_baseline_somalologic_samples.csv)
meta_data	SomaScan meta data produced by prepare_somascan_meta_data() .
transform	Transformation to apply to protein values: • "log2" (default): log base 2 • "log10": log base 10 • "none"
qc	QC steps to apply. • "default" (default): Exclude participant samples with hybridization control scale factor out of range. • "flagged": Exclude participants samples flagged for any normalization acceptance criteria. • "none": Do not exclude any participant samples.
repeat_samples	Default "remove" will remove data for the second sample for participants who had a sample run twice. Use "retain" to keep this data; the second samples will have sample_source=1.
columns	Names of any additional columns from sample_data to include in the output.

QC acceptance criteria

Hybridization control scale factor (hybcontrolnormscale) and median signal normalization scale factors (normscale_20, normscale_0_5, normscale_0_005) have an acceptance range of 0.4-2.5.

Percentage of measurements used for ANML scale factor calculation (anmlfractionused_20, anmlfractionused_0_5, anmlfractionused_0_005) has a criteria of > 30% (0.3).

Samples that meet these criteria are given a PASS for normalization acceptance criteria for all row scale factors (rowcheck), while samples that are outside the criteria are given a FLAG.

Hybridization control scale factor out of range suggests a technical issue. Set qc = "default" to exclude participant samples with hybridization control scale factor (hybcontrolnormscale) out of range (i.e. less than 0.4 or greater than 2.5).

Other criteria out of range suggests that the sample is fairly different from a typical population. Set qc = "flagged" to exclude participant samples with a FLAG for normalization acceptance criteria for all row scale factors (rowcheck).

```
prepare_somascan_meta_data
```

Preparation of SomaScan meta data

Description

Preparation of SomaScan meta data

Usage

```
prepare_somascan_meta_data(
  data,
  names = c("protein", "id"),
  columns = c("seqid", "somaaid", "target_full_name", "target", "uniprot", "organism",
    "type", "dilution"),
  filter = "HumanProtein"
)
```

Arguments

data	Raw CKB SomaScan meta data (imported from somalogic_meta.rds or somalogic_meta.csv)
names	Naming scheme to use for proteins: "protein" (default) to use target name or "id" to use SomaLogic sequence IDs.
columns	Optional columns to include in the meta data. (Default: c("seqid", "somaaid", "target_full_name", "target", "uniprot", "organism", "type", "dilution"))
filter	Filter to apply to rows: • "HumanProtein" (default): Only return meta data for aptamers where organism is "Human" and type is "Protein". • "All": Return all rows of meta data.

`safemodels`*Safe versions of regression model function*

Description

These functions are wrappers around the regression model functions that will capture side-effects such as errors and warnings and return a list. This is particularly useful when using the model functions iteratively.

Usage`linear_model_safely(...)``logistic_model_safely(...)``cox_model_safely(...)`**Arguments**

`...` Arguments are passed to the model function.

Value

A named list:

- `result`: What is returned by the model function.
- `errors`: Errors produced by the model function as a character string.
- `warnings`: Warnings produced by the model function as a character string.
- `messages`: Messages produced by the model function as a character string.
- `output`: Output produced by the model function as a character string.

See Also

[linear_model\(\)](#), [logistic_model\(\)](#), [cox_model\(\)](#)

`summarise_continuous`*Summarise continuous variables*

Description

Summarise continuous variables

Usage`summarise_continuous(data, cols, by = NULL)`

Arguments

data	Data set (e.g. produced by <code>prepare_proteomics_analysis_dataset()</code>)
cols	Names of columns containing continuous variables to be summarised.
by	Names of columns that contain categorical participant characteristics to produce subgroup summaries.

trend	<i>Test for trend</i>
-------	-----------------------

Description

A chi-squared test for trend (on 1 degree of freedom) given a vector of independent estimates and their standard errors.

Usage

```
trend(beta, se)
```

Arguments

beta	vector of independent estimates
se	vector of their standard errors

Details

See `vignette("trend_heterogeneity")` for details.

Value

A named list:

- test statistic (chi-squared test statistic)
- p (p-value)

Index

* datasets

- ckbtools_ascertainments, 2
- ckbtools_baseline, 3
- ckbtools_endpoint_definitions, 3
- ckbtools_endpoints, 3
- ckbtools_participant_data, 4
- ckbtools_raw_olink_meta_data, 4
- ckbtools_raw_olink_protein_data, 4
- ckbtools_raw_somascan_meta_data, 5
- ckbtools_raw_somascan_protein_data, 5
- ckbtools_raw_somascan_sample_data, 5
- ckbtools_ascertainments, 2
- ckbtools_baseline, 3
- ckbtools_endpoint_definitions, 3
- ckbtools_endpoints, 3
- ckbtools_participant_data, 4
- ckbtools_raw_olink_meta_data, 4
- ckbtools_raw_olink_protein_data, 4
- ckbtools_raw_somascan_meta_data, 5
- ckbtools_raw_somascan_protein_data, 5
- ckbtools_raw_somascan_sample_data, 5
- cox_model, 6
- cox_model(), 17, 23
- cox_model_safely (safemodells), 23
- cox_model_safely(), 17
- coxph, 6
- emmeans, 12
- expand_age_at_risk, 8
- float, 7, 12, 14
- ftrend, 14
- glm, 13
- heterogeneity, 9
- incidence_rates, 10
- linear_model, 11
- linear_model(), 17, 23
- linear_model_safely (safemodells), 23
- linear_model_safely(), 17
- lm, 11
- logistic_model, 13
- logistic_model(), 17, 23
- logistic_model_safely (safemodells), 23
- logistic_model_safely(), 17
- multiple_models, 15
- prepare_olink_data, 18
- prepare_olink_data(), 20
- prepare_olink_meta_data, 19
- prepare_olink_meta_data(), 18, 20
- prepare_proteomics_analysis_dataset, 20
- prepare_somascan_data, 21
- prepare_somascan_data(), 20
- prepare_somascan_meta_data, 22
- prepare_somascan_meta_data(), 20, 21
- purrr::pmap(), 17
- qvcalc, 7, 12, 14
- ref_grid, 12
- safemodells, 23
- summarise_continuous, 23
- trend, 24