Package: HealthMarkers (via r-universe)

June 2, 2026

Type Package

Title Toolkit for Clinical, Metabolic, and Cardiovascular Biomarker Calculations

Version 0.1.2

Description Computes over 50 specialist health marker functions covering insulin sensitivity and resistance indices (fasting, oral glucose tolerance test, adipose-tissue, tracer, and dual-energy X-ray absorptiometry (DXA)-based), glycaemic and lipid markers, atherogenic and metabolic syndrome scores, liver steatosis and fibrosis scores, and cardiovascular risk algorithms (Framingham Heart Study equations, atherosclerotic cardiovascular disease (ASCVD) Pooled Cohort Equations, the QRISK3 cardiac risk score, and Systematic Coronary Risk Evaluation 2 (SCORE2) including the Older Persons variant (SCORE2-OP)). Also implements renal function (estimated glomerular filtration rate (eGFR), Kidney Failure Risk Equation (KFRE), chronic kidney disease (CKD) staging), pulmonary function (spirometry z-scores, Body-mass index, airflow Obstruction, Dyspnea, and Exercise capacity index (BODE)), inflammatory markers and the inflammatory age clock (iAge), hormonal panels, body composition and anthropometric z-scores, bone turnover markers and fracture risk (Fracture Risk Assessment Tool (FRAX)), frailty and comorbidity indices (Rockwood, Charlson), psychiatric rating scales, and biomarker panels from alternative biofluids (urine, saliva, sweat). Missing value imputation helpers, pre or post computation normalization and a unified all_health_markers() dispatcher that returns all requested marker groups as a single wide tibble are included.

License MIT + file LICENSE

URL https://sufyansuleman.github.io/HealthMarkers/, https://github.com/sufyansuleman/HealthMarkers

BugReports https://github.com/sufyansuleman/HealthMarkers/issues

Depends R (>= 4.0)

Imports dplyr (>= 1.0.0), rlang (>= 1.0.0), stats, tibble (>= 3.0.0), utils, vctrs, Rdpack

Suggests knitr, rmarkdown, testthat (>= 3.0.0), pkgload, CVrisk, di, rspiro, PooledCohort, QRISK3, RiskScorescvd, usethis, withr, mice, missForest

RdMacros Rdpack

Encoding UTF-8

Language en-US

RoxygenNote 7.3.2

Config/testthat/edition 3

NeedsCompilation no

Author Sufyan Suleman [aut, cre] (ORCID: <https://orcid.org/0000-0001-6612-6915>)

Maintainer Sufyan Suleman <sufyansuleman@hotmail.com>

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

Date/Publication 2026-06-02 01:02:00 UTC

RemoteUrl https://github.com/cranhaven/cranhaven.r-universe.dev

RemoteRef package/HealthMarkers

RemoteSha 37ff6f5d8f588ab200143a2d7d7a950f174e4c46

RemoteSubdir HealthMarkers

Contents

adipo_is 4
adiposity_sds 6
adiposity_sds_strat 8
all_health_markers 9
all_insulin_indices 11
allostatic_load 12
alm_bmi_index 14
asrs_score 15
atherogenic_indices 17
bis_score 18
bode_index 19
bone_markers 20
calc_sds 22
charlson_index 24
ckd_stage 26
cognitive_score 27
corrected_calcium 28
cvd_marker_aip 29
cvd_marker_ldl_particle_number 30

cvd_risk . . . 31
cvd_risk_ascvd . . . 32
cvd_risk_qrisk3 . . . 33
cvd_risk_scorescvd . . . 34
cvd_risk_stroke . . . 35
fasting_is . . . 36
frailty_index . . . 37
frax_score . . . 40
gad7_score . . . 41
ghq12_score . . . 42
glycemic_markers . . . 43
health_summary . . . 46
hm_col_report . . . 47
hm_normalize . . . 48
hormone_markers . . . 49
iAge . . . 51
impute_mice . . . 53
impute_missforest . . . 55
impute_missing . . . 56
inflammatory_markers . . . 57
isi_score . . . 59
k10_score . . . 60
k6_score . . . 61
kidney_failure_risk . . . 63
kyn_trp_ratio . . . 65
lipid_markers . . . 66
liver_fat_markers . . . 68
liver_markers . . . 70
marker_summary . . . 73
mdq_score . . . 73
metabolic_markers . . . 75
metabolic_risk_features . . . 76
metss . . . 78
nfl_marker . . . 80
normalize_vec . . . 82
nutrient_markers . . . 83
obesity_indices . . . 86
ogtt_is . . . 89
oxidative_markers . . . 92
phq9_score . . . 93
plot_frailty_age . . . 94
psych_dx_flags . . . 95
psych_markers . . . 96
psych_med_flags . . . 97
pulmo_markers . . . 98
renal_markers . . . 100
saliva_markers . . . 102
sarc_f_score . . . 105

spirometry_markers 106
spq_score 107
sweat_markers 108
tracer_dxa_is 110
urine_markers 112
validate_inputs 114
vitamin_d_status 115
vitamin_markers 116
who5_score 118

Index **120**

`adipo_is` *Adipose insulin sensitivity indices (QUICKI, VAI, LAP, TyG, TG/HDL, Belfiore)*

Description

Computes adipose-related insulin sensitivity/resistance indices from fasting inputs. Expected input units (converted internally):

- Glucose G0 mmol/L -> mg/dL (* 18)
- Insulin I0 pmol/L -> muU/mL (/ 6)
- TG mmol/L -> mg/dL (* 88.57)
- HDL mmol/L -> mg/dL (* 38.67)

Reported indices (higher magnitude of negative "_inv" values implies worse adipose IR):

- Revised_QUICKI = 1 / (log10(I0 (muU/mL)) + log10(G0 (mg/dL)) + log10(FFA (mmol/L)))
- VAI (sex-specific; inverted as VAI_*_inv so larger negative = worse)
- TG_HDL_C_inv = -(TG/HDL) in mg/dL
- TyG_inv = -ln(TG (mg/dL) * G0 (mg/dL) / 2)
- LAP (sex-specific; inverted)
- McAuley_index = exp(2.63 - 0.28 ln(I0 (muU/mL)) - 0.31 ln(TG (mmol/L)))
- Adipo_inv = -(FFA * I0 (muU/mL))
- Belfiore_inv_FFA = - 2 / (I0 (muU/mL) * FFA + 1)

Inversion Note: Most indices (VAI, LAP, TyG, TG/HDL, Adipo, Belfiore) are algebraically inverted from their original insulin RESISTANCE definitions so that more negative values consistently indicate worse adipose insulin sensitivity. Revised_QUICKI and McAuley_index are retained in their original orientation (already sensitivity indices; higher = better). See the vignette for detailed interpretation guidance.

Usage

```
adipo_is(
  data,
  col_map = NULL,
  normalize = "none",
  na_action = c("keep", "omit", "error"),
  verbose = TRUE,
  ...
)
```

Arguments

`data`	Data frame or tibble with required columns mapped by `col_map`
`col_map`	Named list mapping keys to columns: G0, I0, TG, HDL_c, FFA, waist, bmi
`normalize`	One of c("none","z","inverse","range","robust"); default "none"
`na_action`	One of c("keep","omit","error"); default "keep"
`verbose`	Logical; if `TRUE` (default), prints column mapping, the list of indices being computed, and a per-column results summary.
`...`	Reserved

Value

A tibble with columns: `Revised_QUICKI`, `VAI_Men_inv`, `VAI_Women_inv`, `TG_HDL_C_inv`, `TyG_inv`, `LAP_Men_inv`, `LAP_Women_inv`, `McAuley_index`, `Adipo_inv`, `Belfiore_inv_FFA`. If an ID column is detected in `data` (e.g. `id`, `IID`, `participant_id`), it is prepended as the first output column.

References

Katz A, Nambi SS, Mather K, Baron AD, Follmann DA, Sullivan G, Quon MJ (2000). "Quantitative Insulin Sensitivity Check Index: A Simple, Accurate Method for Assessing Insulin Sensitivity in Humans." *Journal of Clinical Endocrinology & Metabolism*, **85**(7), 2402–2410. doi:10.1210/jcem.85.7.6661. Amato MC, Giordano C, Galia M, Criscimanna A, Vitabile S, Midiri M, Galluzzo A (2010). "Visceral Adiposity Index: A Reliable Indicator of Visceral Fat Function Associated with Cardiometabolic Risk." *Diabetes Care*, **33**(4), 920–922. doi:10.2337/dc091825. Kahn HS (2005). "The Lipid Accumulation Product Performs Better than Body Mass Index as an Indicator of Cardiovascular Risk in Women." *BMC Cardiovascular Disorders*, **5**(1), 26. doi:10.1186/14712261526. Guerrero-Romero F, Simental-Mendía LE, González-Ortiz M, Martínez-Abundis E, Ramos-Zavala MG, Hernández-González SO, Jacques-Camarena O, Rodríguez-Morán M (2010). "The Product of Triglycerides and Glucose, a Simple Measure of Insulin Sensitivity. Comparison with the Euglycemic-Hyperinsulinemic Clamp." *Journal of Clinical Endocrinology & Metabolism*, **95**(7), 3347–3351. doi:10.1210/jc.20100288. Dobiášová M, Frohlich JJ (2001). "The Plasma Parameter Log(TG/HDL-C) as an Atherogenic Index: Correlation with Lipoprotein Particle Size and Esterification Rate in ApoB-Lipoprotein-Depleted Plasma." *Clinical Biochemistry*, **34**(7), 583–588. doi:10.1016/S00099120(01)002636. Belfiore F, Iannello S, Volpicelli G (1998). "Insulin Sensitivity Indices Calculated from Basal and OGTT-Related Insulin and Glucose Levels." *Molecular Genetics and Metabolism*, **63**(2), 134–141. doi:10.1006/mgme.1997.2658. Raynaud E, Pérez-Martin

A, Brun J, Benhaddad AA, Mercier J (1999). "Fasting Plasma Insulin and Insulin Resistance Indices." *Diabetes & Metabolism*, **25**(6), 524–532. No DOI identified in Crossref/PubMed as of 2026-03-16; see URL, https://pubmed.ncbi.nlm.nih.gov/?term=Fasting+Plasma+Insulin+and+Insulin+Resistance+Indices.

Examples

```
df <- tibble::tibble(
  G0 = c(5.2, 6.1),      # mmol/L
  I0 = c(60, 110),       # pmol/L
  TG = c(1.2, 1.8),      # mmol/L
  HDL_c = c(1.3, 1.0),   # mmol/L
  FFA = c(0.4, 0.6),     # mmol/L
  waist = c(85, 102),    # cm
  bmi = c(24, 31)        # kg/m^2
)
cm <- as.list(names(df)); names(cm) <- names(df)
out <- adipo_is(df, cm, verbose = FALSE, na_action = "keep")
```

`adiposity_sds` *Calculate standardized scores (SDS) for adiposity measures*

Description

Computes standard deviation (z) scores for anthropometric variables relative to a single (non-sex-stratified) reference set of means and standard deviations. Includes input validation, optional raw-value extreme screening/capping, configurable handling of extreme SDS values, NA row policies, optional concise summary output, and optional verbose progress messages.

Usage

```
adiposity_sds(
  data,
  col_map = NULL,
  ref,
  na_action = c("keep", "omit", "error"),
  extreme_action = c("cap", "NA", "error", "warn", "ignore"),
  sds_cap = 6,
  diagnostics = FALSE,
  warn_thresholds = list(na_prop = 0.05, extreme_prop = 0.01),
  id_col = NULL,
  return_summary = FALSE,
  verbose = TRUE,
  na_strategy = NULL,
  extreme_strategy = NULL
)
```

Arguments

data	data.frame or tibble containing the measurement columns.
col_map	Optional named list mapping reference variable names to column names in data. If NULL, identity mapping is assumed (names(ref) must be in data).
ref	Named list where each element is a numeric vector with names mean and sd, e.g. list(BMI = c(mean = 23, sd = 4)).
na_action	One of c("keep","omit","error") for row handling when any mapped variable is missing.
extreme_action	One of c("cap","NA","error","warn","ignore") for SDS values exceeding sds_cap.
sds_cap	Positive numeric; absolute cap used when extreme_action = "cap".
diagnostics	Logical; if TRUE emit informational/warning messages (coercions, missingness, extremes). FALSE suppresses non-critical warnings.
warn_thresholds	Named list with optional elements na_prop (default 0.05) and extreme_prop (default 0.01) used for proportion-based diagnostic warnings.
id_col	Optional column name used only in verbose summaries.
return_summary	Logical; if TRUE return a list with elements data, summary, and warnings instead of just the SDS tibble.
verbose	Logical; if TRUE print progress and completion summaries.
na_strategy	Soft-deprecated alias for na_action (if provided and na_action missing, it is used).
extreme_strategy	Soft-deprecated alias for extreme_action (if provided and extreme_action missing, it is used).

Details

SDS are computed as: (observed - mean) / sd. Rows are removed only when na_action = 'omit'. Raw-value extreme screening (if enabled) is applied before SDS computation. Extreme SDS handling (cap / warn / error / ignore) is controlled by extreme_action. Legacy argument aliases (na_strategy, extreme_strategy) are soft-deprecated but still accepted.

Value

A tibble with one <var>_SDS column per reference variable, or a list when return_summary = TRUE.

Examples

```
ref <- list(BMI = c(mean = 23, sd = 4), waist = c(mean = 80, sd = 12))
df <- data.frame(BMI = c(25, NA, 60, 18), waist = c(85, 70, 300, 55))
adiposity_sds(df, ref)
```

`adiposity_sds_strat` *Compute sex-stratified standardized scores (SDS) for adiposity measures*

Description

Computes sex-specific SDS (z-scores) for selected anthropometric variables using reference means and SDs provided separately for males and females.

Usage

```
adiposity_sds_strat(
  data,
  col_map,
  ref,
  na_action = c("keep", "omit", "error"),
  allow_partial = FALSE,
  prefix = "",
  verbose = TRUE
)
```

Arguments

`data`	Data frame or tibble with variables and a sex column
`col_map`	Named list mapping: • sex: column name with sex values ("M","F","m","f", 1, 2) • vars: optional named list mapping reference variable names -> data column names. If omitted, expects the reference variable names to exist in `data`.
`ref`	Named list with elements "M" and "F". Each is a named list of numeric vectors c(mean=, sd=) keyed by variable name, e.g.: list(M = list(BMI = c(mean=23, sd=3.5), waist = c(mean=85, sd=10)), F = list(BMI = c(mean=21, sd=3.0), waist = c(mean=75, sd=9)))
`na_action`	One of: • "keep" - keep rows with NA (propagates to outputs) • "omit" - drop rows with NA in any required variable • "error" - abort if any required variable has NA
`allow_partial`	If TRUE, skip variables absent in data (with a warning); if FALSE error
`prefix`	Optional prefix for output columns (default "")
`verbose`	Logical; when TRUE emit progress via package logger; by default logging is controlled by options(healthmarkers.verbose)

Value

A tibble with one SDS column per retained variable: `varname_SDS`, where `varname` is the original variable name (optionally prefixed by `prefix`).

References

World Health Organization (1995). *Physical Status: The Use and Interpretation of Anthropometry*, volume 854 of *Technical Report Series*. World Health Organization. ISBN 9241208546, No DOI for this WHO report; see ISBN/URL, https://www.who.int/publications/i/item/9241208546.

Examples

```
ref <- list(
  M = list(BMI = c(mean=24.5, sd=3.8), waist = c(mean=88, sd=12)),
  F = list(BMI = c(mean=22.1, sd=4.2), waist = c(mean=76, sd=11))
)
df <- data.frame(BMI=c(25.2,21.8,27.1), waist=c(85,72,95), sex=c("M","F","M"))
adiposity_sds_strat(
  df,
  col_map = list(sex = "sex", vars = list(BMI = "BMI", waist = "waist")),
  ref = ref
)
```

---

all_health_markers *Compute all available HealthMarkers categories*

Description

Compute all available HealthMarkers categories

Usage

```
all_health_markers(
  data,
  col_map,
  which = "all",
  include_insulin = TRUE,
  normalize = c("none", "z", "inverse", "range", "robust"),
  mode = c("both", "IS", "IR"),
  verbose = TRUE,
  na_action = c("keep", "omit", "error"),
  id_col = NULL,
  return_input = TRUE
)
```

Arguments

data	A data.frame or tibble.
col_map	Named list for column mapping forwarded to underlying functions. If col_map is NULL or an empty list, all_health_markers() calls hm_col_report() once at the top level to guess a column map from common synonyms (for example

TG vs triglycerides, BMI vs bmi, HDL_c vs HDL). The inferred col_map is then reused for all groups that require it, and an error is thrown if required keys (e.g. TG, HDL_c, LDL_c, TC, BMI, age, sex) cannot be inferred.

which — "all" or a vector of registry keys (see Details).

include_insulin — Logical; include all_insulin_indices() first.

normalize — One of c("none","z","inverse","range","robust").

mode — One of c("both","IS","IR") passed to insulin indices.

verbose — Logical.

na_action — One of c("keep","omit","error"); forwarded to underlying calculators (HM-CS v2).

id_col — Optional character string naming a column in data to include in the returned output when return_input = FALSE. Ignored when return_input = TRUE. Typical use: participant ID or sample barcode.

return_input — Logical (default TRUE). When TRUE the original input columns are retained in the output (current behaviour). When FALSE only the newly computed marker columns are returned, plus id_col if supplied. Set to FALSE to avoid carrying a large input data frame through the pipeline — join back later with cbind() or dplyr::left_join() on id_col.

Details

Common group names for which include:

- "lipid", "liver", "glycemic", "mets", "oxidative"
- "bone", "allostatic_load", "nutrient", "vitamin", "vitamin_d_status"
- "renal", "ckd_stage", "kidney_kfre"
- "frailty_index", "charlson", "sarc_f"
- "nfl", "iAge", "calcium_corrected", "kyn_trp"

Value

Data frame. When return_input = TRUE (default): original columns plus all derived markers. When return_input = FALSE: only the newly computed columns (and id_col if specified).

Note

For academic / clinical references tied to each derived marker or index, consult the help pages of the source functions (e.g. ?allostatic_load, ?bone_markers, ?vitamin_markers, ?inflammatory_markers, etc.). This aggregator provides integration only and does not restate citations.

Aggregator wrapper. See underlying function help pages for full references across categories included by which.

Examples

```
# Quick smoke-test (lipid group only, no insulin)
df <- data.frame(TC = 200, HDL_c = 50, TG = 150, LDL_c = 120)
all_health_markers(df, col_map = list(), which = "lipid",
                   include_insulin = FALSE, normalize = "none",
                   verbose = FALSE, na_action = "keep")


# Lipid + liver groups
df <- data.frame(
  TC = 200, HDL_c = 50, TG = 150, LDL_c = 120,
  ALT = 30, AST = 20, BMI = 25
)
all_health_markers(df, col_map = list(), which = c("lipid","liver"),
                   include_insulin = FALSE, normalize = "none", mode = "both",
                   verbose = FALSE, na_action = "keep")
```

---

all_insulin_indices *Compute insulin sensitivity/resistance panels (fasting, OGTT, adipose, tracer/DXA)*

Description

Compute insulin sensitivity/resistance panels (fasting, OGTT, adipose, tracer/DXA)

Usage

```
all_insulin_indices(
  data,
  col_map = NULL,
  normalize = c("none", "z", "inverse", "range", "robust"),
  mode = c("both", "IS", "IR"),
  verbose = TRUE,
  na_action = c("keep", "omit", "error")
)
```

Arguments

data	A data.frame or tibble of raw measurements.
col_map	Named list with keys G0,I0,G30,I30,G120,I120,TG,HDL_c,FFA,waist,weight,bmi,age,sex,rate_palmitate
normalize	One of c("none","z","inverse","range","robust").
mode	One of c("IS","IR","both"). "IR" returns only inverted IR, "IS" only the original IS, "both" returns both with IR_ prefix.
verbose	Logical.
na_action	One of c("keep","omit","error"); forwarded to underlying calculators (HM-CS v2).

Value

A tibble of IS (and/or IR_) columns.

Note

For scholarly references to specific indices (e.g., HOMA-IR, QUICKI, Raynaud, Belfiore, tracer-derived indices, adiposity-related IS metrics), consult the individual function help pages (e.g. ?fasting_is, ?ogtt_is, ?adipo_is, ?tracer_dxa_is). Citations are intentionally not duplicated here.

Aggregator wrapper. See underlying function help pages for full references: fasting_is(), ogtt_is(), adipo_is(), tracer_dxa_is().

References

Suleman S, Madsen AL, Ängquist LH, Schubert M, Linneberg A, Loos RJF, Hansen T, Grarup N (2024). "Genetic Underpinnings of Fasting and Oral Glucose-stimulated Based Insulin Sensitivity Indices." *The Journal of Clinical Endocrinology & Metabolism*, **109**(11), 2754–2763. doi:10.1210/clinem/dgae275.

Examples

```
# Quick smoke-test (fasting indices only)
df <- data.frame(G0 = 5.2, I0 = 60)
all_insulin_indices(df, normalize = "none", mode = "IS",
                    verbose = FALSE, na_action = "keep")


# Full panel with all supported inputs
df <- data.frame(
  G0 = 5.2, I0 = 60, G30 = 7.5, I30 = 90, G120 = 6.2, I120 = 80,
  TG = 1.5, HDL_c = 1.3, FFA = 0.3, waist = 85, weight = 70, bmi = 24,
  age = 40, sex = "M", rate_palmitate = 0.1, rate_glycerol = 0.2, fat_mass = 20
)
all_insulin_indices(df, col_map = list(
  G0="G0", I0="I0", G30="G30", I30="I30", G120="G120", I120="I120",
  TG="TG", HDL_c="HDL_c", FFA="FFA", waist="waist", weight="weight",
  bmi="bmi", age="age", sex="sex", rate_palmitate="rate_palmitate",
  rate_glycerol="rate_glycerol", fat_mass="fat_mass"
), normalize = "none", mode = "IS", verbose = FALSE, na_action = "keep")
```

---

allostatic_load *Allostatic Load Index*

Description

Computes a composite Allostatic Load (AL) score by flagging biomarkers that exceed user-specified high-risk thresholds (strict > when multiple biomarkers; inclusive >= when only one biomarker). Aligned with HM-CS v3: structured validation, diagnostic control, verbose reporting, and optional summary output.

Usage

```
allostatic_load(
  data,
  thresholds,
  col_map = NULL,
  na_action = c("keep", "omit", "error"),
  return_summary = FALSE,
  verbose = TRUE
)
```

Arguments

`data`	data.frame or tibble of numeric biomarker columns.
`thresholds`	named list of scalar numeric cutoffs (names must match columns).
`col_map`	optional named list mapping keys in `thresholds` to column names in `data`.
`na_action`	one of c("keep","omit","error") ("keep" treats NA as zero contribution).
`return_summary`	logical; TRUE returns list(data, summary, warnings).
`verbose`	logical; print progress messages via hm_inform() (also gated by options(healthmarkers.verbose)).

Details

API pattern note: Unlike most HealthMarkers functions which follow the standard `(data, col_map, ...)` signature, `allostatic_load()` uses `(data, thresholds, col_map = NULL, ...)` because `thresholds` is a domain-specific required argument that cannot be inferred from column names alone. As a result, this function is not directly dispatchable through the standard `all_health_markers()` registry; a custom wrapper that supplies `thresholds` explicitly would be required.

Value

tibble with AllostaticLoad or list when return_summary = TRUE.

References

Seeman TE, Singer BH, Rowe JW, Horwitz RI, McEwen BS (1997). "Price of Adaptation—Allostatic Load and Its Health Consequences." *Archives of Internal Medicine*, **157**(19), 2259–2268. doi:10.1001/archinte.1997.00440400111013.

See Also

`adiposity_sds`, `adiposity_sds_strat`

Examples

```
df <- tibble::tibble(
  SBP = c(118, 142, 130),
  DBP = c(76, 92, 85),
  CRP = c(1.2, 4.8, 2.1)
)
```

```
thr <- list(SBP = 130, DBP = 85, CRP = 3)
allostatic_load(df, thresholds = thr, na_action = "keep", verbose = FALSE)

# Single biomarker uses inclusive >= rule
allostatic_load(df, thresholds = list(CRP = 3))
```

`alm_bmi_index` *Appendicular Lean Mass to BMI Index*

Description

Calculates the ratio of appendicular lean mass (ALM) to body mass index (BMI), and flags low muscle mass based on FNIH Sarcopenia Project cut-points.

Usage

```
alm_bmi_index(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error", "ignore", "warn"),
  verbose = TRUE
)
```

Arguments

`data`	A data.frame or tibble with ALM, BMI, and sex columns.
`col_map`	Named list with: • alm: appendicular lean mass column name (kg) • bmi: body mass index column name (kg/m^2) • sex: sex column name ("Male"/"Female" or m/f; case-insensitive)
`na_action`	One of c("keep","omit","error","ignore","warn").
`verbose`	Logical; if TRUE (default), emits progress messages.

Details

ALM/BMI reflects muscle mass relative to body size. FNIH cut-points:

- Men: ALM/BMI < 0.789
- Women: ALM/BMI < 0.512

ALM should be in kilograms and BMI in kg/m^2.

Value

A tibble with:

- alm_bmi_ratio (numeric)
- low_muscle_mass (logical; TRUE if below sex-specific cut-point; NA if sex unknown or ratio NA)

References

McLean RR, Shardell MD, Alley DE, et al. (2014). "Criteria for clinically relevant weakness and low lean mass: FNIH Sarcopenia Project." *Journal of Gerontology A: Biological Sciences and Medical Sciences*, **69**(5), 576–583. doi:10.1093/gerona/glu012.

Examples

```
df <- data.frame(ALM_kg = c(7.2, 5.8, 6.5), BMI = c(24, 28, 22),
                 Sex = c("male", "female", "male"))
alm_bmi_index(df)
```

---

asrs_score *Adult ADHD Self-Report Scale scoring*

Description

Adult ADHD Self-Report Scale scoring

Usage

```
asrs_score(
  data,
  col_map = list(),
  na_action = c("keep", "omit", "error"),
  missing_prop_max = 0.2,
  impute = c("none", "mean"),
  prefix = "ASRS",
  partA_items = sprintf("asrs_%02d", 1:6),
  partA_thresholds = c(3, 3, 3, 4, 4, 4),
  partA_cutoff = 4,
  verbose = TRUE
)
```

Arguments

data	Data frame containing questionnaire item columns.
col_map	Named list mapping canonical item IDs to column names; defaults assume items are already named.
na_action	How to handle rows with missing items: keep, omit, or error.

`missing_prop_max` Maximum allowed proportion of missing items per row before the score is set to `NA`.

`impute` Imputation strategy for missing items when under the threshold: `none` or `mean` (row-wise mean).

`prefix` Prefix for output column names.

`partA_items` Vector of Part A item IDs.

`partA_thresholds` Numeric thresholds applied to Part A items.

`partA_cutoff` Count threshold for Part A positivity.

`verbose` Logical; if `TRUE`, emits informational messages about column resolution and scoring progress via `hm_inform()`.

Value

A tibble of score columns only: `ASRS_total`, `ASRS_partA_count`, `ASRS_partA_positive`. Input columns are not included.

Note

Items are expected on a 0–4 scale (Never=0, Rarely=1, Sometimes=2, Often=3, Very Often=4). The default `partA_thresholds` follow the official WHO ASRS v1.1 guide: items 1–3 are positive at ≥ 3 (Often or Very Often) and items 4–6 at ≥ 4 (Very Often only).

References

Kessler RC, Adler LA, Ames M, Demler O, Faraone SV, Hiripi E, Howes MJ, Jin R, Secnik K, Spencer T, Üstün TB, Walters EE (2005). "The World Health Organization Adult ADHD Self-Report Scale (ASRS): A Short Screening Scale for Use in the General Population." *Psychological Medicine*, **35**(2), 245–256. doi:10.1017/S0033291704002892. Kessler RC, Adler L, Ames M, Demler O, Faraone SV, Hiripi E, Howes MJ, Jin R, Secnik K, Spencer T, Ustün TB, Walters EE (2006). "The Prevalence and Correlates of Adult ADHD in the United States: Results from the National Comorbidity Survey Replication." *American Journal of Psychiatry*, **163**(4), 716–723. doi:10.1176/ajp.2006.163.4.716. (prevalence study; ASRS used as instrument)

Examples

```
df <- data.frame(matrix(2, nrow = 1, ncol = 18))
names(df) <- sprintf("asrs_%02d", 1:18)
asrs_score(df)
```

`atherogenic_indices` *Compute atherogenic indices*

Description

Calculates:

- AIP: Atherogenic Index of Plasma = log10(TG / HDL_c)
- CRI_I: Castelli Risk Index I = TC / HDL_c
- CRI_II: Castelli Risk Index II = LDL_c / HDL_c

Usage

```
atherogenic_indices(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error"),
  normalize = c("none", "log10"),
  verbose = TRUE
)
```

Arguments

`data`	data.frame/tibble with lipid columns.
`col_map`	named list mapping keys to columns, e.g. list(TG="TG", HDL_c="HDL_c", TC="TC", LDL_c="LDL_c").
`na_action`	one of c("keep","omit","error").
`normalize`	one of c("none","log10"). Reserved; AIP always uses log10(TG/HDL_c).
`verbose`	Logical; if `TRUE` (default), prints column mapping, the list of indices being computed, and a per-column results summary.

Details

Behavior:

- Required keys: TG, HDL_c. Optional: TC, LDL_c.
- NA policy via `na_action`: "keep" (default), "omit" (drop rows with any NA in used lipids), "error".
- Extreme screening via `check_extreme` and `extreme_action` ("warn","cap","error","ignore","NA"). Default bounds (mg/dL) used only for screening: TG (0, 10000), HDL_c (0, 1000), LDL_c (0, 10000), TC (0, 10000). Note: All indices are unitless ratios; units cancel in computations.
- Emits progress via `hm_inform()` when `verbose = TRUE` or when package option enables logs.

Value

tibble with columns AIP, CRI_I, CRI_II. If an ID column is detected in `data` (e.g. `id`, `IID`, `participant_id`), it is prepended.

References

Dobiášová M (2004). "Atherogenic Index of Plasma (AIP) log(Triglycerides/HDL-Cholesterol): Theoretical and Practical Implications." *Clinical Chemistry*, **50**(7), 1113–1115. doi:10.1373/clinchem.2004.033175.
Castelli WP, Doyle JT, Gordon T, et al. (1977). "High-Density Lipoprotein Cholesterol and Other Lipids in Coronary Heart Disease: The Framingham Study." *American Journal of Medicine*, **62**(5), 707–714. doi:10.1016/00029343(77)908749.

Examples

```
df <- tibble::tibble(
  TG = c(150, 200),
  HDL_c = c(50, 40),
  TC = c(200, 220),
  LDL_c = c(120, 150)
)
cm <- list(TG = "TG", HDL_c = "HDL_c", TC = "TC", LDL_c = "LDL_c")
atherogenic_indices(df, col_map = cm)
```

---

bis_score *Barratt Impulsiveness Scale (key-driven)*

Description

Barratt Impulsiveness Scale (key-driven)

Usage

```
bis_score(
  data,
  col_map = list(),
  key,
  na_action = c("keep", "omit", "error"),
  missing_prop_max = 0.2,
  impute = c("none", "mean"),
  prefix = "BIS",
  verbose = TRUE
)
```

Arguments

`data`	Data frame containing questionnaire item columns.
`col_map`	Named list mapping canonical item IDs to column names; defaults assume items are already named.
`key`	List with `items`, `min_val`, `max_val`, optional `reverse` and `subscales`.
`na_action`	How to handle rows with missing items: `keep`, `omit`, or `error`.
`missing_prop_max`	Maximum allowed proportion of missing items per row before the score is set to `NA`.
`impute`	Imputation strategy for missing items when under the threshold: `none` or `mean` (row-wise mean).
`prefix`	Prefix for output column names.
`verbose`	Logical; if `TRUE`, emits informational messages about column resolution and scoring progress via `hm_inform()`.

Value

A tibble of score columns only (total and optional subscales). Input columns are not included.

References

Patton JH, Stanford MS, Barratt ES (1995). "Factor Structure of the Barratt Impulsiveness Scale." *Journal of Clinical Psychology*, **51**(6), 768–774. doi:10.1002/10974679(199511)51:6<768::AID-JCLP2270510607>3.0.CO;21.

Examples

```
bis_key <- list(items = sprintf("bis_%02d", 1:5), min_val = 1, max_val = 4)
df <- data.frame(bis_01 = 1, bis_02 = 2, bis_03 = 3, bis_04 = 4, bis_05 = 2)
bis_score(df, key = bis_key)
```

`bode_index` *BODE Index (BMI, Obstruction, Dyspnea, Exercise capacity)*

Description

Computes the BODE index (0-10) using FEV1 % predicted, 6-minute walk distance (6MWD), mMRC dyspnea scale, and BMI. Higher scores indicate worse prognosis in COPD.

Usage

```
bode_index(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error", "ignore", "warn"),
  verbose = TRUE
)
```

Arguments

data	data.frame/tibble with required columns.
col_map	named list with keys: fev1_pct OR (fev1 and fev1_pred) OR fev1_pp; plus sixmwd, mmrc, bmi. Example minimal: list(fev1_pct="FEV1pct", sixmwd="Walk_m", mmrc="mMRC", bmi="BMI") Example derive: list(fev1="FEV1", fev1_pred="FEV1_pred", sixmwd="Walk_m", mmrc="mMRC", bmi="BMI") Example from spirometry_markers: list(fev1_pp="fev1_pp", sixmwd="Walk_m", mmrc="mMRC", bmi="BMI")
na_action	one of c("keep","omit","error","ignore","warn").
verbose	logical; TRUE (default) emits messages.

Details

Scoring components: FEV1 % predicted: >=65 = 0; 50-64 = 1; 36-49 = 2; <=35 = 3 6MWD (meters): >=350 = 0; 250-349 = 1; 150-249 = 2; <=149 = 3 mMRC dyspnea: 0-1 = 0; 2 = 1; 3 = 2; 4 = 3 BMI: >21 = 0; <=21 = 1

Value

tibble with bode_index (integer). NA if any required input missing (unless omitted).

References

Celli BR, Cote CG, Marin JM, Casanova C, Montes de Oca M, Mendez R, Pinto-Plata V, Cabral HJ (2004). "The BODE Index in Chronic Obstructive Pulmonary Disease." *The New England Journal of Medicine*, **350**, 1005–1012. doi:10.1056/NEJMoa021322.

Examples

```
df <- data.frame(FEV1pct = c(68, 45, 30), Walk_m = c(400, 280, 140),
                 mMRC = c(1, 2, 3), BMI = c(24, 19, 18))
bode_index(df, col_map = list(fev1_pct = "FEV1pct", sixmwd = "Walk_m",
                              mmrc = "mMRC", bmi = "BMI"))
```

---

bone_markers *Compute Bone Health & Body-Composition Markers (HM-CS v3)*

Description

Given DXA, anthropometry, and optional bone-turnover markers, computes:

- OSTA: (weight - age) x 0.2
- ALMI: Appendicular Lean Mass Index = ALM / height^2
- FMI: Fat Mass Index = FM / height^2
- BMD_Tscore: (BMD - ref_mean) / ref_sd and (if in col_map + data) passes through: TBS, HSA, PINP, CTX, BSAP, Osteocalcin.

Usage

```
bone_markers(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error"),
  verbose = TRUE
)
```

Arguments

data	A data.frame or tibble with subject-level DXA/anthropometry data.
col_map	Named list mapping keys to column names. Required keys: • age, weight, height, ALM, FM, BMD, BMD_ref_mean, BMD_ref_sd Optional (passed-through if present and found in data): TBS, HSA, PINP, CTX, BSAP, Osteocalcin.
na_action	One of "keep", "omit", or "error" controlling how missing/non-finite input values are treated.
verbose	Logical; if TRUE (default), emits progress messages via hm_inform().

Details

Notes:

- Units: height in meters; ALM, FM, weight in kilograms; BMD in g/cm^2; ALMI/FMI in kg/m^2.
- Non-finite values are treated as NA; division by zero is prevented by input checks.
- BMD_ref_mean and BMD_ref_sd must be supplied by the user from an appropriate reference population (for example, study-specific values or external norms such as NHANES).

Value

A tibble with columns: OSTA, ALMI, FMI, BMD_Tscore, and optionally TBS, HSA, PINP, CTX, BSAP, Osteocalcin (in that order).

References

Koh LK, Ben Sedrine W, Torralba TP, Kung A, others (2001). "A Simple Tool to Identify Asian Women at Increased Risk of Osteoporosis." *Osteoporosis International*, **12**(8), 699–705. doi:10.1007/s001980170070. World Health Organization (1994). *Assessment of Fracture Risk and Its Application to Screening for Postmenopausal Osteoporosis*, volume 843 of *Technical Report Series*. World Health Organization. No DOI for this WHO report; see URL, https://iris.who.int/handle/10665/39142.

Examples

```
library(tibble)
df <- tibble(
  age = c(60, 72), weight = c(65, 50), height = c(1.65, 1.58),
  ALM = c(18.2, 14.7), FM = c(22.0, 20.5),
  BMD = c(0.95, 0.80), BMD_ref_mean = c(1.00, 1.00), BMD_ref_sd = c(0.12, 0.12)
)
col_map <- list(
  age = "age", weight = "weight", height = "height",
  ALM = "ALM", FM = "FM", BMD = "BMD",
  BMD_ref_mean = "BMD_ref_mean", BMD_ref_sd = "BMD_ref_sd"
)
bone_markers(df, col_map)
```

---

calc_sds *Calculate Standard Deviation Scores (SDS; z-scores) for health markers*

Description

Computes per-variable SDS as (x - mean) / sd using supplied reference statistics. Includes input validation, NA/error handling, data quality warnings, and verbose progress via the package logger (hm_inform), aligned with HM-CS v3.

Usage

```
calc_sds(
  data,
  vars,
  ref,
  id_col = NULL,
  sds_cap = 6,
  na_strategy = c("omit", "error", "keep"),
  extreme_strategy = c("cap", "warn", "error", "NA"),
  warn_thresholds = list(na_prop = 0.05, extreme_prop = 0.01),
  return = c("data", "list"),
  verbose = TRUE,
  na_action = NULL,
  check_extreme = TRUE,
  extreme_action = NULL
)
```

Arguments

data	A data.frame/tibble containing the variables.
vars	Character vector of variable names in data to compute SDS for. Must be non-empty and unique.

`ref` A data.frame with columns: `variable`, `mean`, `sd` supplying reference statistics for each variable listed in `vars`. Must contain exactly one row per requested variable, with finite `mean` and `sd > 0`.

`id_col` Optional character scalar naming an ID column in `data` (used only in messages and duplicate-ID notes).

`sds_cap` Numeric scalar; absolute cap for SDS when `extreme_strategy = "cap"`. Must be positive. Default 6.

`na_strategy` One of c("omit","error","keep"):

- "omit": drop rows with missing values across any of `vars` (default)
- "error": stop if any missing values among `vars`
- "keep": keep rows; SDS for missing inputs will be NA

`extreme_strategy`

One of c("cap","warn","error","NA"):

- "cap": cap |SDS| at `sds_cap` and warn (default)
- "warn": keep extreme SDS, but warn
- "error": stop if any |SDS| > `sds_cap`
- "NA": set extreme SDS to NA

`warn_thresholds`

Named list controlling warnings (proportions in $[0, 1]$):

- na_prop: warn if proportion of rows with NA among `vars` exceeds this (default 0.05)
- extreme_prop: warn if proportion of extreme SDS (cells) exceeds this (default 0.01)

`return` One of c("data","list"). "data" returns a tibble with added `[var]_sds` columns (default). "list" returns a list with components `data`, `summary`, and `warnings` (backward compatible).

`verbose` Logical; if TRUE, emit progress via hm_inform(). Also controlled by options(healthmarkers.verbose = "none"|"inform"|"debug").

`na_action` Optional HM-CS alias for `na_strategy` (accepted values: keep/omit/error; used if provided).

`check_extreme` Logical; if TRUE, enables SDS extreme handling (alias for legacy behavior; defaults to TRUE in this implementation).

`extreme_action` Optional HM-CS alias for `extreme_strategy` (accepted values: cap/NA/error; used if provided).

Details

Derivation: for each requested variable, SDS is calculated as `(observed_value - reference_mean) / reference_sd` using user-supplied reference statistics in `ref`.

Usage note: this function does not derive normative reference values internally. Users should provide `ref` from the target population (preferred), or from an external source matched as closely as possible on age/sex/ethnicity/context. Interpretability depends on the quality and relevance of those supplied reference means and SDs.

By default, returns a tibble with added `[var]_sds` columns (tidyverse-friendly). For backward compatibility, you can request the previous list output.

Value

- If `return = "data"` (default): a tibble with added `[var]_sds` columns.
- If `return = "list"`: a list with:
  - data: tibble with added SDS columns
  - summary: list with input/output row counts, omitted rows, total extremes, and per-variable summary
  - warnings: character vector of warning messages emitted

Examples

```
ref <- data.frame(
  variable = c("bmi","sbp"),
  mean     = c(25, 120),
  sd       = c(4, 15)
)
df <- data.frame(
  id  = 1:6,
  bmi = c(24, 30, NA, 29, 10, 26),
  sbp = c(118, 200, 119, 121, 500, 120)
)
out_tbl <- calc_sds(
  data = df,
  vars = c("bmi","sbp"),
  ref = ref,
  id_col = "id",
  na_strategy = "omit",
  extreme_strategy = "cap",
  sds_cap = 6,
  verbose = FALSE
)
```

---

charlson_index *Charlson Comorbidity Index (CCI)*

Description

Computes the Charlson Comorbidity Index by summing weighted comorbidities.

Usage

```
charlson_index(
  data,
  col_map = list(mi = "mi", chf = "chf", pvd = "pvd", stroke = "stroke", dementia =
    "dementia", copd = "copd", rheum = "rheum", ulcer = "ulcer", mild_liver =
    "mild_liver", diabetes = "diabetes", diab_comp = "diab_comp", hemiplegia =
   "hemiplegia", renal = "renal", cancer = "cancer", leukemia = "leukemia", lymphoma =
   "lymphoma", sev_liver = "sev_liver", metastatic_cancer = "metastatic_cancer", hiv =
```

```
    "hiv"),
  verbose = TRUE,
  na_action = c("keep", "omit", "error", "ignore", "warn")
)
```

Arguments

data	A data.frame or tibble with binary indicators (0/1) for each comorbidity.
col_map	Named list mapping keys to columns in data: mi, chf, pvd, stroke, dementia, copd, rheum, ulcer, mild_liver, diabetes, diab_comp, hemiplegia, renal, cancer, leukemia, lymphoma, sev_liver, metastatic_cancer, hiv.
verbose	Logical; if TRUE, emits progress messages.
na_action	One of c("keep","omit","error","ignore","warn").

Details

The Charlson Index predicts 10-year mortality by summing weighted comorbidities. We implement the canonical 19-condition, weight scheme:

- 1 point: myocardial infarction, congestive heart failure, peripheral vascular disease, cerebrovascular disease (stroke), dementia, chronic pulmonary disease (COPD), rheumatologic disease, peptic ulcer disease
- 2 points: hemiplegia/paraplegia, moderate/severe renal disease, any malignancy (non-metastatic), leukemia, lymphoma, diabetes with complications
- 3 points: moderate/severe liver disease
- 6 points: metastatic solid tumor, AIDS/HIV

To avoid double counting paired conditions, the following use the maximum applicable weight:

- Diabetes: max(1 * diabetes without complications, 2 * diabetes with complications)
- Liver disease: max(1 * mild liver disease, 3 * moderate/severe liver disease)
- Cancer: max(2 * non-metastatic solid tumor, 6 * metastatic solid tumor)

Age points are not included here and can be added separately if needed.

Value

A tibble with one column: charlson_index (integer total score; NA if any required input is NA and na_action != "omit").

References

Charlson ME, Pompei P, Ales KL, MacKenzie CR (1987). "A New Method of Classifying Prognostic Comorbidity in Longitudinal Studies: Development and Validation." *Journal of Chronic Diseases*, **40**(5), 373–383. doi:10.1016/00219681(87)901718.

Examples

```
patient <- tibble::tibble(
  mi=0, chf=0, pvd=0, stroke=0, dementia=0, copd=0, rheum=0, ulcer=0,
  mild_liver=0, diabetes=0, diab_comp=1, hemiplegia=0, renal=1,
  cancer=0, leukemia=0, lymphoma=0, sev_liver=0, metastatic_cancer=0, hiv=0
)
charlson_index(
  patient,
  col_map = as.list(stats::setNames(names(patient), names(patient)))
)
```

---

ckd_stage *CKD staging (GFR and albuminuria) and KDIGO risk*

Description

Categorizes eGFR into G1-G5, albuminuria into A1-A3 (by UACR mg/g), and maps KDIGO risk.

Usage

```
ckd_stage(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error"),
  verbose = TRUE
)
```

Arguments

`data`	Data frame with renal measures.
`col_map`	Named list with required key: eGFR; optional key: UACR.
`na_action`	One of: • "keep" (retain rows; stages become NA where inputs missing) • "omit" (drop rows with any missing eGFR/UACR that are mapped) • "error" (abort if any mapped input missing)
`verbose`	Logical; if TRUE (default), emits progress messages via `hm_inform()`.

Value

Tibble with CKD_stage, Albuminuria_stage, KDIGO_risk.

References

Kidney Disease: Improving Global Outcomes (KDIGO) CKD Work Group (2013). "KDIGO 2012 Clinical Practice Guideline for the Evaluation and Management of Chronic Kidney Disease." *Kidney International Supplements*, **3**(1), 1–150. doi:10.1038/kisup.2012.73, Related synopsis: Stevens and Levin (2013), Ann Intern Med, doi:10.7326/0003-4819-158-11-201306040-00007, https://kdigo.org/guidelines/ckd-evaluation-and-management/.

Examples

```
df <- data.frame(eGFR = c(95, 50), UACR = c(10, 200))
ckd_stage(df, list(eGFR = "eGFR", UACR = "UACR"))
```

`cognitive_score` *Cognitive composite (z-mean or PCA1)*

Description

Cognitive composite (z-mean or PCA1)

Usage

```
cognitive_score(
  data,
  col_map = list(),
  na_action = c("keep", "omit", "error"),
  missing_prop_max = 0.2,
  method = c("z_mean", "pca1"),
  prefix = "cog",
  verbose = TRUE
)
```

Arguments

`data`	Data frame containing questionnaire item columns.
`col_map`	Named list with `tasks` mapping task IDs to column names (>= 2 tasks required).
`na_action`	How to handle rows with missing items: `keep`, `omit`, or `error`.
`missing_prop_max`	Maximum allowed proportion of missing items per row before the score is set to `NA`.
`method`	Aggregation method: `z_mean` (average of z-scores) or `pca1` (first PC).
`prefix`	Prefix for output column names.
`verbose`	Logical; if `TRUE`, emits informational messages about column resolution and scoring progress via `hm_inform()`.

Value

A tibble of score columns only: `{prefix}_z_mean` or `{prefix}_pca1`. Input columns are not included.

Examples

```
df <- data.frame(task_a = c(1, 2), task_b = c(2, 3), task_c = c(3, 4))
cm <- list(tasks = list(
  task_a = "task_a",
  task_b = "task_b",
  task_c = "task_c"
))
cognitive_score(df, col_map = cm, method = "z_mean")
```

---

corrected_calcium *Albumin-Corrected Calcium*

Description

Calculates the albumin-adjusted (corrected) serum calcium level, accounting for hypoalbuminemia, using the Payne formula.

Usage

```
corrected_calcium(
  data,
  col_map = NULL,
  units = c("auto", "conventional", "si"),
  na_action = c("keep", "omit", "error", "ignore", "warn"),
  verbose = TRUE
)
```

Arguments

data	A data.frame or tibble containing serum calcium and albumin.
col_map	Named list with calcium and albumin indicating column names.
units	One of c("auto","conventional","si"). "auto" attempts unit detection.
na_action	One of c("keep","omit","error","ignore","warn").
verbose	Logical; if TRUE (default), emits progress via hm_inform().

Details

Payne formula (conventional units): Corrected Ca (mg/dL) = measured Ca (mg/dL) + 0.8 * (4.0 - albumin (g/dL)). If inputs appear to be in SI units (calcium mmol/L, albumin g/L), they are converted to mg/dL and g/dL (using 1 mmol/L ~= 4 mg/dL; 1 g/L = 0.1 g/dL) for the correction and converted back to mmol/L for output.

Value

A tibble with one column: corrected_calcium (numeric, in mg/dL for conventional input or mmol/L for SI / auto-SI input).

References

Payne RB, Little AJ, Williams RB, Milner JR (1973). "Interpretation of serum calcium in patients with abnormal serum proteins." *British Medical Journal*, **4**(5893), 643–646. doi:10.1136/bmj.4.5893.643.

Examples

```
df <- data.frame(Ca = c(2.3, 2.5, 2.1), Alb = c(38, 42, 30))
corrected_calcium(df)
```

---

cvd_marker_aip *Atherogenic Index of Plasma (AIP)*

Description

Computes log10(TG / HDL_c) with input validation and HM-CS NA handling.

Usage

```
cvd_marker_aip(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error"),
  verbose = TRUE
)
```

Arguments

`data`	A data frame with numeric columns TG and HDL_c (mg/dL).
`col_map`	Named list mapping required keys: • TG: triglycerides • HDL_c: HDL cholesterol
`na_action`	One of: • "keep" (retain rows; AIP is NA where inputs missing/non-finite) • "omit" (drop rows with any missing/non-finite inputs) • "error" (abort if any required input missing/non-finite)
`verbose`	Logical; if TRUE, emit hm_inform() progress messages.

Value

A tibble with columns model = "AIP" and value.

References

Dobiášová M (2004). "Atherogenic Index of Plasma (AIP) log(Triglycerides/HDL-Cholesterol): Theoretical and Practical Implications." *Clinical Chemistry*, **50**(7), 1113–1115. doi:10.1373/clinchem.2004.033175.

Examples

```
df <- data.frame(TG = c(150, 200), HDL_c = c(50, 40))
cvd_marker_aip(df)
```

`cvd_marker_ldl_particle_number` *LDL Particle Number Estimate (via ApoB)*

Description

Returns ApoB as a proxy for LDL particle number with HM-CS NA handling.

Usage

```
cvd_marker_ldl_particle_number(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error"),
  verbose = TRUE
)
```

Arguments

`data`	A data frame with numeric column ApoB (mg/dL).
`col_map`	Named list mapping required key: • ApoB
`na_action`	One of: • "keep" (retain rows; value is NA where input missing/non-finite) • "omit" (drop rows with missing/non-finite input) • "error" (abort if input missing/non-finite)
`verbose`	Logical; if TRUE, emit hm_inform() progress messages.

Value

A tibble with columns model = "LDL_PN" and value.

Examples

```
df <- data.frame(ApoB = c(80, 120, 100))
cvd_marker_ldl_particle_number(df)
```

`cvd_risk` *Compute cardiovascular risk or marker by selected model*

Description

Dispatch to the appropriate risk or marker function, or run all of them. Includes basic argument validation and robust fallback to NA rows if individual calculators fail.

Usage

```
cvd_risk(
  data,
  model = c("ALL", "ASCVD", "QRISK3", "Stroke", "RiskScorescvd", "AIP", "LDL_PN"),
  year = 10,
  ...,
  verbose = TRUE
)
```

Arguments

`data`	Data frame required by your chosen model.
`model`	One of: • "ALL" • Risk calculators: "ASCVD","QRISK3","Stroke","RiskScorescvd" • Lipid markers: "AIP","LDL_PN"
`year`	Risk horizon (10 or 30) for applicable models; ignored for lipid markers.
`...`	Forwarded to underlying wrappers (e.g., col_map, na_action).
`verbose`	Logical; if TRUE, prints progress (legacy; messages now routed via hm_inform).

Value

A tibble.

Examples

```
df <- data.frame(TG = c(150, 200), HDL_c = c(50, 40))
cvd_risk(df, model = "AIP")
```

`cvd_risk_ascvd` *ASCVD risk (ACC/AHA Pooled Cohort Equations)*

Description

Wrapper around the PooledCohort ASCVD calculators with added input validation, optional data-quality warnings, and quiet failure to NA if the backend errors.

Usage

```
cvd_risk_ascvd(
  data,
  year = 10,
  col_map = NULL,
  na_warn_prop = 0.2,
  verbose = TRUE,
  ...
)
```

Arguments

`data`	A data frame with the required cardiovascular risk columns.
`year`	Risk horizon: 10 or 30.
`col_map`	Optional named list mapping internal keys (`age`, `sex`, `race`, `smoker`, `total_chol`, `HDL_c`, `sbp`, `bp_treated`, `diabetes`, `bmi`) to actual column names in `data`. If `NULL` (default), column names are auto-inferred then fall back to the key names themselves. `sex` accepts `1`/`0`, `"m"`/`"f"`, or `"male"`/`"female"` (case-insensitive).
`na_warn_prop`	Proportion (0-1) to flag high missingness warnings (default 0.2). Only used when `verbose = TRUE`; underlying backend handles NA as per its API.
`verbose`	Logical; if TRUE, prints progress and a short summary.
`...`	Passed to the underlying PooledCohort function.

Value

A tibble with columns `model`, `year`, `risk` (percentage).

References

Goff DC, Lloyd-Jones DM, Bennett G, Coady S, D'Agostino RB, et al. (2014). "2013/2014 ACC/AHA Guideline on the Assessment of Cardiovascular Risk." *Circulation*, **129**(25 Suppl 2), S49–S73. doi:10.1161/01.cir.0000437741.48606.98, Pooled Cohort Equations; ACC/AHA Task Force on Practice Guidelines.

Examples

```
df <- tibble::tibble(
  age = 55, sex = 1, race = "white", smoker = FALSE,
  total_chol = 200, HDL_c = 50, sbp = 140, bp_treated = FALSE,
  diabetes = FALSE, bmi = 27
)
if (requireNamespace("PooledCohort", quietly = TRUE)) {
  cvd_risk_ascvd(df, year = 10, verbose = TRUE)
}
```

---

cvd_risk_qrisk3 *QRISK3 10-year risk (UK QRISK3-2017)*

Description

Wrapper around `QRISK3::QRISK3_2017()` that auto-generates a `patid` if one is not supplied. Adds input validation and quiet failure to NA on backend error.

Usage

```
cvd_risk_qrisk3(data, ..., patid = NULL, na_warn_prop = 0.2, verbose = TRUE)
```

Arguments

`data`	A data frame with variables required by `QRISK3::QRISK3_2017()`.
`...`	Passed to `QRISK3::QRISK3_2017()`.
`patid`	Optional vector of patient IDs (default: `1:nrow(data)`).
`na_warn_prop`	Proportion (0-1) to flag high missingness warnings (default 0.2). Only used when `verbose = TRUE`.
`verbose`	Logical; if TRUE, prints progress and a short summary.

Value

A tibble with columns `model`, `year`, `risk`.

References

Hippisley-Cox J, Coupland C, Brindle P, et al. (2017). "Development and validation of QRISK3 risk prediction algorithms." *BMJ*, **357**, j2099. doi:10.1136/bmj.j2099.

Examples

```
if (requireNamespace("QRISK3", quietly = TRUE)) {
  df <- data.frame(
    gender = 1L, age = 55L,
    atrial_fibrillation = 0L, atypical_antipsy = 0L,
    regular_steroid_tablets = 0L, erectile_disfunction = 0L,
    migraine = 0L, rheumatoid_arthritis = 0L,
    chronic_kidney_disease = 0L, severe_mental_illness = 0L,
    systemic_lupus_erythematosis = 0L, blood_pressure_treatment = 0L,
    diabetes1 = 0L, diabetes2 = 0L,
    weight = 80, height = 175,
    ethnicity = 1L, heart_attack_relative = 0L,
    cholesterol_HDL_ratio = 4.2, systolic_blood_pressure = 130,
    std_systolic_blood_pressure = 7, smoke = 0L, townsend = 0
  )
  cvd_risk_qrisk3(df)
}
```

---

cvd_risk_scorescvd *RiskScorescvd calculator*

Description

Passthrough to `RiskScorescvd::calc_scores()` with graceful fallback to NA.

Usage

```
cvd_risk_scorescvd(data, ...)
```

Arguments

`data`	Data required by `RiskScorescvd::calc_scores()`.
`...`	Passed to `RiskScorescvd::calc_scores()`.

Value

Object returned by `RiskScorescvd::calc_scores()`.

Examples

```
if (requireNamespace("RiskScorescvd", quietly = TRUE)) {
  df <- data.frame(
    Age = 55, Sex = 0, Smoking_status = 1,
    systolic.bp = 140, Total_cholesterol = 5.5,
    HDL.cholesterol = 1.3
  )
  cvd_risk_scorescvd(df)
```

```
}
```

`cvd_risk_stroke` *Stroke 10-year risk*

Description

Wrapper around `PooledCohort::predict_10yr_stroke_risk()` with quiet fallback to NA if the backend errors.

Usage

```
cvd_risk_stroke(data, col_map = NULL, na_warn_prop = 0.2, verbose = TRUE, ...)
```

Arguments

`data`	A data frame with the required cardiovascular risk columns.
`col_map`	Optional named list mapping internal keys (`age`, `sex`, `race`, `smoker`, `total_chol`, `HDL_c`, `sbp`, `bp_treated`, `diabetes`, `bmi`) to actual column names in `data`. If `NULL` (default), column names are auto-inferred then fall back to the key names themselves. `sex` accepts `1/0`, `"m"/"f"`, or `"male"/"female"` (case-insensitive).
`na_warn_prop`	Proportion (0-1) to flag high missingness warnings (default 0.2). Only used when `verbose = TRUE`.
`verbose`	Logical; if TRUE, prints progress and a short summary.
`...`	Passed to `PooledCohort::predict_10yr_stroke_risk()`.

Value

A tibble with `model`, `year`, `risk`.

References

Goff DC, Lloyd-Jones DM, Bennett G, Coady S, D'Agostino RB, et al. (2014). "2013/2014 ACC/AHA Guideline on the Assessment of Cardiovascular Risk." *Circulation*, **129**(25 Suppl 2), S49–S73. doi:10.1161/01.cir.0000437741.48606.98, Pooled Cohort Equations; ACC/AHA Task Force on Practice Guidelines.

Examples

```
if (requireNamespace("PooledCohort", quietly = TRUE)) {
  df <- data.frame(age = 55, sex = 1, race = "white", smoker = FALSE,
    total_chol = 200, HDL_c = 50, sbp = 140, bp_treated = FALSE,
    diabetes = FALSE, bmi = 27)
  cvd_risk_stroke(df)
}
```

`fasting_is` *Calculate fasting-based insulin sensitivity indices*

Description

Compute 10 fasting indices from glucose (mmol/L) and insulin (pmol/L): Fasting_inv, Raynaud, HOMA_IR_inv, FIRI, QUICKI, Belfiore_basal, Ig_ratio_basal, Isi_basal, Bennett, HOMA_IR_rev_inv. Units converted internally: G0_mg = G0*18 (mg/dL), I0_u = I0/6 (muU/mL).

Usage

```
fasting_is(
  data,
  col_map = NULL,
  normalize = c("none", "z", "inverse", "range", "robust"),
  na_action = c("keep", "omit", "error", "warn"),
  verbose = TRUE
)
```

Arguments

`data`	Data frame with required inputs.
`col_map`	Named list mapping required keys: • G0: fasting glucose (mmol/L) • I0: fasting insulin (pmol/L)
`normalize`	One of: "none","z","inverse","range","robust".
`na_action`	One of: • "keep" (retain all rows; indices become NA where inputs missing/non-finite) • "omit" (drop rows with any missing/non-finite required inputs) • "error" (abort if any required input is missing/non-finite) • "warn" (emit a warning for rows with missing inputs, then keep them)
`verbose`	Logical; if `TRUE` (default), prints column mapping, the list of indices being computed, and a per-column results summary.

Value

Tibble with 10 columns (indices listed above). If an ID column is detected in `data` (e.g. `id`, `IID`, `participant_id`), it is prepended as the first output column.

References

Matthews DR, Hosker JP, Rudenski AS, Naylor BA, Treacher DF, Turner RC (1985). "Homeostasis Model Assessment: Insulin Resistance and Beta-Cell Function from Fasting Plasma Glucose and Insulin Concentrations in Man." *Diabetologia*, **28**(7), 412–419. doi:10.1007/BF00280883. Katz A, Nambi SS, Mather K, Baron AD, Follmann DA, Sullivan G, Quon MJ (2000). "Quantitative Insulin Sensitivity Check Index: A Simple, Accurate Method for Assessing Insulin Sensitivity in Humans." *Journal of Clinical Endocrinology & Metabolism*, **85**(7), 2402–2410. doi:10.1210/jcem.85.7.6661. Raynaud E, Pérez-Martin A, Brun J, Benhaddad AA, Mercier J (1999). "Fasting Plasma Insulin and Insulin Resistance Indices." *Diabetes & Metabolism*, **25**(6), 524–532. No DOI identified in Crossref/PubMed as of 2026-03-16; see URL, https://pubmed.ncbi.nlm.nih.gov/?term=Fasting+Plasma+Insulin+and+Insulin+Resistance+Indices. Avignon A, Charles M, Rabasa-Lhoret R, et al. (1999). "Assessment of Insulin Sensitivity from Oral Glucose Tolerance Test in Normal Subjects and in Insulin-Resistant Patients." *International Journal of Obesity*, **23**(5), 512–517. doi:10.1038/sj.ijo.0800864. Belfiore F, Iannello S, Volpicelli G (1998). "Insulin Sensitivity Indices Calculated from Basal and OGTT-Related Insulin and Glucose Levels." *Molecular Genetics and Metabolism*, **63**(2), 134–141. doi:10.1006/mgme.1997.2658. Sluiter D, Erkelens DW, Reitsma WD, Doorenbos H (1976). "Glucose Tolerance and Insulin Release: A Mathematical Approach." *Diabetes*, **25**, 245–249. doi:10.2337/diabetes.25.4.245. Hanson RL, Pratley RE, Bogardus C, Narayan KMV, Roumain J, Imperatore G, Fagot-Campagna A, Pettitt DJ, Bennett PH, Knowler WC (2000). "Evaluation of Simple Indices of Insulin Sensitivity and Insulin Secretion for Use in Epidemiologic Studies." *American Journal of Epidemiology*, **151**(2), 190–198. doi:10.1093/oxfordjournals.aje.a010187. Anderson RL, Hamman RF, Savage PJ, Saad MF, Laws A, Kades WW, Sands RE, Cefalu WT (1995). "Exploration of Simple Measures of Insulin Resistance." *American Journal of Epidemiology*, **142**(7), 724–732. doi:10.1093/aje/142.7.724.

Examples

```
# Minimal example (units: G0 in mmol/L, I0 in pmol/L)
df <- data.frame(G0 = c(5.2, 6.1, 4.8), I0 = c(60, 120, 80))
res <- fasting_is(df, col_map = list(G0 = "G0", I0 = "I0"))
head(res)

# With NA handling
df2 <- data.frame(G0 = c(5.0, NA), I0 = c(90, 150))
fasting_is(df2, col_map = list(G0 = "G0", I0 = "I0"), na_action = "keep")
```

`frailty_index` *Compute Frailty (Deficit) Index using di::di with QA and verbose summaries*

Description

Thin wrapper around the di package's di() that:

- Validates inputs and arguments.
- Coerces tibbles to base data.frames (for di()'s class checks).
- Auto-selects numeric deficit columns when cols = NULL (excluding age if supplied).

- Optionally scans for missing/out-of-range values with warnings or errors.
- Provides step-by-step verbose output and a completion summary.
- Optionally returns a tidy tibble instead of the original list.

Usage

```
frailty_index(
  data,
  cols = NULL,
  invert = NULL,
  rescale = TRUE,
  age = NULL,
  rescale.custom = NULL,
  rescale.avoid = NULL,
  bins = 7,
  visible = FALSE,
  na_action = c("ignore", "warn", "error", "keep", "omit"),
  na_warn_prop = 0.2,
  return = c("list", "data"),
  verbose = TRUE
)
```

Arguments

`data`	A data.frame or tibble of health deficits (ideally binary/logical or scaled to $[0, 1]$). Non-binary numeric columns can be rescaled by di::di when rescale = TRUE.
`cols`	Character vector of deficit column names to use. If NULL (default), all numeric columns are used except age (if supplied).
`invert`	Character vector of column names whose values should be inverted by di::di (e.g., where higher values indicate better health).
`rescale`	Logical; if TRUE, non-binary columns will be rescaled to $[0, 1]$ by di::di. Default TRUE.
`age`	Optional name of the column holding age (used by di for plotting and optional age-binned outputs; excluded from auto-selected cols).
`rescale.custom`	Advanced argument passed through to di::di. See di::di documentation for syntax.
`rescale.avoid`	Advanced argument passed through to di::di; see di::di documentation for syntax.
`bins`	Integer; number of age bins for FI-by-age plots. Default 7.
`visible`	Logical; if TRUE and age is provided, di will draw a plot (via plot.di()). Default FALSE.
`na_action`	One of `c("keep","omit","error","warn","ignore")`. Controls handling of missing values in selected deficit columns. `"keep"` (and its backward-compatible alias `"ignore"`) passes NAs through to di::di. `"warn"` emits a warning and then keeps NAs (alias for `"keep"` with a missingness warning). `"omit"` drops rows

with any NA in selected deficits before computing. `"error"` stops if any NA is detected. Default `"ignore"` (retained for backward compatibility; equivalent to `"keep"`).

`na_warn_prop` Proportion in $[0, 1]$ above which a high-missingness warning is emitted (per column) when na_action = "warn". Default 0.2.

`return` One of c("list","data"). "list" (default) returns the original di::di result (backward compatible). "data" returns a tibble with one row per individual, columns: di (the frailty index) plus the selected deficit columns (post-capping if applied). Age is included if present.

`verbose` Logical; if TRUE, prints progress and a completion summary.

Details

Background The Frailty Index (FI) is computed as the proportion of health deficits present in an individual across a set of candidate deficits. The approach was introduced and formalized by Rockwood and Mitnitski and subsequently standardized for construction and reporting.

Value

- If return = "list" (default): the object returned by di::di (typically a list with di and columns).
- If return = "data": a tibble with di and the selected columns.

References

Mitnitski AB, Mogilner AJ, Rockwood K (2001). "Accumulation of Deficits as a Proxy Measure of Aging." *The Scientific World Journal*, **1**, 323–336. doi:10.1100/tsw.2001.58. Rockwood K, Mitnitski A (2007). "Frailty in Relation to the Accumulation of Deficits." *Journals of Gerontology Series A*, **62**(7), 722–727. doi:10.1093/gerona/62.7.722. Searle SD, Mitnitski A, Gahbauer EA, Gill TM, Rockwood K (2008). "A Standard Procedure for Creating a Frailty Index." *BMC Geriatrics*, **8**, 24. doi:10.1186/14712318824.

Examples

```
# Minimal example (runs only if the 'di' package is installed)
if (requireNamespace("di", quietly = TRUE)) {
  df <- data.frame(
    age = c(70, 75, 80),
    d1 = c(0, 1, 1),
    d2 = c(0.2, 0.8, 1.0),
    d3 = c(TRUE, FALSE, TRUE)
  )
  # Auto-select numeric deficits; returns list (di, columns)
  res <- frailty_index(df, cols = NULL, age = "age", verbose = TRUE)
  # Tidy tibble return
  tb  <- frailty_index(df, cols = c("d1","d2","d3"), age = "age",
                       return = "data", verbose = TRUE)
}
```

frax_score *FRAX 10-Year Fracture Risk Score (simplified placeholder)*

Description

Estimates the 10-year probabilities of major osteoporotic and hip fracture using a simplified, non-validated approximation based on FRAX risk factors. This is for development/demo only and does not implement the proprietary FRAX algorithm.

Usage

```
frax_score(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error", "ignore", "warn"),
  country = NULL,
  verbose = TRUE
)
```

Arguments

data	A data frame or tibble with inputs.
col_map	Named list mapping required and optional inputs: • Required: age, sex • Optional binary risk factors: prior_fracture, parent_fracture, steroids, rheumatoid, secondary_op, smoker, alcohol • Optional: bmd (T-score; if present and in (-5,0], used to adjust risk)
na_action	One of c("keep","omit","error","ignore","warn").
country	optional country/region code for FRAX calibration (accepted, currently unused).
verbose	Logical; if TRUE, emits progress via rlang::inform.

Details

Important: this function is an educational placeholder and must not be used for clinical decision-making, patient counseling, or guideline-based treatment selection.

Value

Tibble with frax_major_percent and frax_hip_percent.

Examples

```
df <- data.frame(Age = c(65, 72, 58), Sex = c("female", "female", "male"))
frax_score(df)
```

gad7_score *GAD-7 scoring*

Description

GAD-7 scoring

Usage

```
gad7_score(
  data,
  col_map = list(),
  na_action = c("keep", "omit", "error"),
  missing_prop_max = 0.2,
  impute = c("none", "mean"),
  prefix = "GAD7",
  verbose = TRUE
)
```

Arguments

data	Data frame containing questionnaire item columns.
col_map	Named list mapping canonical item IDs to column names; defaults assume items are already named.
na_action	How to handle rows with missing items: keep, omit, or error.
missing_prop_max	Maximum allowed proportion of missing items per row before the score is set to NA.
impute	Imputation strategy for missing items when under the threshold: none or mean (row-wise mean).
prefix	Prefix for output column names.
verbose	Logical; if TRUE, emits informational messages about column resolution and scoring progress via hm_inform().

Value

A tibble of score columns only: GAD7_total and GAD7_severity (factor). Input columns are not included.

References

Spitzer RL, Kroenke K, Williams JBW, Löwe B (2006). "A Brief Measure for Assessing Generalized Anxiety Disorder: The GAD-7." *Archives of Internal Medicine*, **166**(10), 1092–1097. doi:10.1001/archinte.166.10.1092. Plummer F, Manea L, Trepel D, McMillan D (2016). "Screening for Anxiety Disorders with the GAD-7 and GAD-2: A Systematic Review and Diagnostic Meta-Analysis." *General Hospital Psychiatry*, **39**, 24–31. doi:10.1016/j.genhosppsych.2015.11.005. (validation meta-analysis)

Examples

```
df <- data.frame(gad7_01 = 0, gad7_02 = 1, gad7_03 = 2, gad7_04 = 1,
                 gad7_05 = 0, gad7_06 = 1, gad7_07 = 2)
gad7_score(df)
```

---

ghq12_score *GHQ-12 scoring (Likert or binary)*

Description

GHQ-12 scoring (Likert or binary)

Usage

```
ghq12_score(
  data,
  col_map = list(),
  na_action = c("keep", "omit", "error"),
  missing_prop_max = 0.2,
  impute = c("none", "mean"),
  prefix = "GHQ12",
  method = c("likert", "binary"),
  case_cutoff_binary = 3,
  verbose = TRUE
)
```

Arguments

data	Data frame containing questionnaire item columns.
col_map	Named list mapping canonical item IDs to column names; defaults assume items are already named.
na_action	How to handle rows with missing items: keep, omit, or error.
missing_prop_max	Maximum allowed proportion of missing items per row before the score is set to NA.
impute	Imputation strategy for missing items when under the threshold: none or mean (row-wise mean).
prefix	Prefix for output column names.
method	Scoring method: likert (0-3 per item) or binary (0/1 per item).
case_cutoff_binary	Cut-off for case status when using binary scoring.
verbose	Logical; if TRUE, emits informational messages about column resolution and scoring progress via hm_inform().

Value

A tibble of score columns only: `GHQ12_total_likert` (likert method) or `GHQ12_total_binary` and `GHQ12_case_binary` (binary method). Input columns are not included.

References

Goldberg DP, Williams P (1988). *A User's Guide to the General Health Questionnaire*. NFER-Nelson, Windsor, UK.

Examples

```
df <- data.frame(ghq12_01 = 0, ghq12_02 = 1, ghq12_03 = 2, ghq12_04 = 1,
                 ghq12_05 = 0, ghq12_06 = 1, ghq12_07 = 0, ghq12_08 = 1,
                 ghq12_09 = 2, ghq12_10 = 1, ghq12_11 = 0, ghq12_12 = 1)
ghq12_score(df, method = "likert")
```

---

glycemic_markers *Calculate glycemic-, C-peptide-, and additional metabolic markers*

Description

Given fasting labs and anthropometry, computes:

- SPISE (Single-Point Insulin Sensitivity Estimator)
- METS_IR (Metabolic Score for Insulin Resistance)
- prediabetes flag (HbA1c >= 42 mmol/mol)
- diabetes flag (HbA1c >= 48 mmol/mol)
- HOMA_CP (C-peptide-based HOMA-IR variant; operational formula, see notes)
- LAR (Leptin/Adiponectin Ratio)
- ASI (Adiponectin Sensitivity Index; adiponectin/insulin)
- TyG_index (Triglyceride-Glucose Index)

Usage

```
glycemic_markers(
  data,
  col_map = NULL,
  na_action = c("ignore", "warn", "error", "keep", "omit"),
  na_warn_prop = 0.2,
  verbose = TRUE
)
```

Arguments

data	A data.frame or tibble containing at least: • HDL_c (mmol/L), TG (mmol/L), BMI (kg/m^2) Optional if present: glucose (mmol/L), HbA1c (mmol/mol), C_peptide (pmol/L), G0 (mmol/L), I0 (pmol/L), leptin (ng/mL), adiponectin (ng/mL).
col_map	Optional named list mapping keys to column names in data. Required keys: HDL_c, TG, BMI. Optional keys (if present will be used): glucose, HbA1c, C_peptide, G0, I0, leptin, adiponectin. If NULL (default), the function uses columns named exactly as the keys.
na_action	One of c("ignore","warn","error","keep","omit"). HM-CS aliases: keep == ignore; omit drops rows with any NA/non-finite in used inputs before computing markers. Default "ignore".
na_warn_prop	Proportion (0-1) threshold for high-missingness warnings among used columns when na_action = "warn". Default 0.2.
verbose	Logical; if TRUE (default), prints step-by-step progress including column mapping, optional input availability, pre-computation notes, physiological range information (informational only, values are not altered), the list of markers being computed with their inputs, and a per-column results summary.

Details

Assumed units (no automatic conversion of inputs except where noted):

- HDL_c, TG: mmol/L (TyG internally converts TG to mg/dL via 88.57)
- BMI: kg/m^2
- glucose, G0: mmol/L (TyG internally converts glucose to mg/dL via 18)
- HbA1c: mmol/mol
- C_peptide, I0: pmol/L (HOMA_CP uses I-like conversion factor 6 as in insulin muU/mL; see notes)
- leptin, adiponectin: ng/mL

These indices are intended for research and feature-engineering applications. The prediabetes and diabetes flags apply standard HbA1c cut-offs (WHO/IDF criteria) but this function is not validated as a clinical diagnostic tool.

Quality controls and options:

- Input validation ensures required variables exist and are numeric-coercible.
- Non-numeric inputs are coerced to numeric with a warning (NAs introduced reported).
- Missing or non-finite inputs are handled via na_action.
- Logs and divisions are computed safely (non-positive arguments yield NA).
- Physiological range notes are printed when verbose = TRUE (values are not altered).
- Verbose mode prints step-by-step progress and a completion summary.

Optional marker detection and pre-computation (one level deep): When `col_map = NULL` (default), column names are inferred automatically. The seven optional keys (glucose, HbA1c, C_peptide, G0, I0, leptin, adiponectin) are computed only when present in `data`. If `BMI` is absent but `weight` (kg) and `height` (m or cm) are present, BMI is computed automatically. If `glucose` is absent but `G0` column exists (or vice versa), the missing key is derived via alias. With `verbose = TRUE` (default), the function reports: column mapping, what is missing and which raw inputs could provide it, which indices will be NA and why, any physiological range notes, and a per-column results summary.

Notes on HOMA_CP:

- This function retains the package's existing operational formula: HOMA_CP = (G0 (mmol/L) * (C_peptide (pmol/L) / 6)) / 22.5 which mirrors HOMA-IR's structure using a 6 pmol/muU scaling used for insulin. Users should verify unit conventions for their datasets; alternative C-peptide HOMA implementations exist (e.g., HOMA2-CP).

Value

A tibble with columns:

- SPISE, METS_IR, prediabetes, diabetes, HOMA_CP, LAR, ASI, TyG_index If an ID column is detected in `data` (e.g. `id`, `IID`, `participant_id`), it is prepended as the first output column.

References

Paulmichl K, Hatunic M, Höbaus C, et al. (2016). "Modification and Validation of the Triglyceride-to–HDL Cholesterol Ratio as a Surrogate of Insulin Sensitivity in White Juveniles and Adults without Diabetes Mellitus: The Single Point Insulin Sensitivity Estimator (SPISE)." *Clinical Chemistry*, **62**(9), 1211–1219. doi:10.1373/clinchem.2016.257436. Bello-Chavolla OY, Almeda-Valdes P, García-Sánchez A, et al. (2018). "METS-IR, a novel score to evaluate insulin sensitivity, is predictive of visceral adiposity and incident type 2 diabetes." *European Journal of Endocrinology*, **178**(5), 533–544. doi:10.1530/EJE170883. Frühbeck G, Catalan V, Rodríguez A, Ramón Sánchez-Recalde Á, Becerril S, Sánchez-González Á, Baena N, Valentí-Azcárate F, Burrell MA, Salvador J (2019). "Adiponectin-leptin Ratio is a Functional Biomarker of Adipose Tissue Inflammation." *Nutrients*, **11**(2), 454. doi:10.3390/nu11020454. Matthews DR, Hosker JP, Rudenski AS, Naylor BA, Treacher DF, Turner RC (1985). "Homeostasis Model Assessment: Insulin Resistance and Beta-Cell Function from Fasting Plasma Glucose and Insulin Concentrations in Man." *Diabetologia*, **28**(7), 412–419. doi:10.1007/BF00280883. Simental-Mendía LE, Rodríguez-Morán M, Guerrero-Romero F (2008). "The Product of Fasting Glucose and Triglycerides as Surrogate for Identifying Insulin Resistance." *Metabolic Syndrome and Related Disorders*, **6**(4), 299–304. doi:10.1089/met.2008.0034. Furler SM, Gan SK, Poynten AM, Chisholm DJ, Campbell LV, Kriketos AD (2006). "Relationship of Adiponectin with Insulin Sensitivity in Humans, Independent of Lipid Availability." *Obesity*, **14**(2), 228–234. doi:10.1038/oby.2006.29.

Examples

```
# Quick smoke-test
df <- data.frame(glucose = 5.6, HbA1c = 44, G0 = 5.5, I0 = 60,
                 HDL_c = 1.2, TG = 1.5, BMI = 24)
glycemic_markers(df, verbose = FALSE)
```

```
df <- tibble::tibble(
  HDL_c       = c(1.0, 1.3),
  TG          = c(1.3, 2.0),
  BMI         = c(24, 30),
  glucose     = c(5.6, 7.1),
  HbA1c       = c(44, 38),
  C_peptide   = c(300, 500),
  G0          = c(5.5, 6.2),
  I0          = c(60, 120),
  leptin      = c(10, 20),
  adiponectin = c(8, 5)
)
glycemic_markers(df)
glycemic_markers(df, verbose = FALSE)
glycemic_markers(df, na_action = "omit", verbose = FALSE)
```

`health_summary` *Summarize selected numeric marker columns*

Description

Lightweight summary for numeric columns: n, n_na, mean, sd, median, p25, p75.

Usage

```
health_summary(x, cols = NULL)
```

Arguments

`x`	A data.frame or tibble.
`cols`	Optional character vector of column names to summarize. If NULL, all numeric columns are summarized.

Value

A tibble with one row per summarized column.

Examples

```
df <- data.frame(a = c(1, 2, NA), b = c(3, 4, 5), c = factor("x"))
health_summary(df)
```

`hm_col_report` *Diagnose and report column name mapping for your data*

Description

Scans the column names of your data frame against the internal synonym dictionary used by all **HealthMarkers** functions and prints a formatted report showing which internal keys were matched automatically and which were not found. The function uses five matching layers in order:

1. **User-supplied** (via `col_map`) — always wins.
2. **Exact synonym match** — column name is in the synonym list.
3. **Case-insensitive exact** — same, ignoring upper/lower case.
4. **Column contains synonym** — data column name contains a synonym as a whole word (e.g.\ `"trig_baseline"` matches `"trig"`).
5. **Synonym contains column** — a synonym contains the column name as a whole word (e.g.\ `"TG_fasting"` synonym matches column `"TG"`).
6. **Fuzzy** (opt-in via `fuzzy = TRUE`) — Levenshtein-based approximate matching as a last resort.

The function returns the matched mappings invisibly as a named list that can be passed directly as the `col_map` argument to any **HealthMarkers** function.

Usage

```
hm_col_report(
  data,
  col_map = NULL,
  verbose = TRUE,
  fuzzy = FALSE,
  show_unmatched = FALSE
)
```

Arguments

`data`	A `data.frame` or `tibble`.
`col_map`	Optional named list of manually specified mappings (`list(TG = "trig_my_col")`). These always take priority.
`verbose`	Logical (default `TRUE`). Print the formatted report.
`fuzzy`	Logical (default `FALSE`). Enable fuzzy / approximate matching as a final layer. May produce false positives; review results with care.
`show_unmatched`	Logical (default `FALSE`). If `TRUE`, also list every key that could not be matched. Useful when you expect a variable to be found but it isn't.

Value

Invisibly returns a named list: internal key → matched data-column name. Keys that were not found are omitted. You can assign this to `col_map` and pass it to any **HealthMarkers** function.

Examples

```
## Not run:
# Basic diagnostic
hm_col_report(my_data)

# Fuzzy matching + show all unmatched
hm_col_report(my_data, fuzzy = TRUE, show_unmatched = TRUE)

# Capture the result as a ready-to-use col_map
cm <- hm_col_report(my_data, verbose = FALSE)
all_health_markers(my_data, col_map = cm)

## End(Not run)
```

---

hm_normalize *Normalise marker columns in a data frame*

Description

A convenience wrapper around `normalize_vec()` that applies a normalisation method to every selected numeric column in a data frame and returns the modified data frame. The most common use-case is to normalise the output of any HealthMarkers function — especially domain functions such as `glycemic_markers()`, `lipid_markers()`, or `renal_markers()` whose internal `normalize` argument currently has no effect.

Usage

```
hm_normalize(
  data,
  cols = NULL,
  method = c("z", "inverse", "range", "robust"),
  skip_cols = NULL,
  ...
)
```

Arguments

`data`	A data frame (or tibble) containing marker columns to normalise.
`cols`	Character vector of column names to normalise. If `NULL` (default), **all numeric columns** in `data` are normalised.
`method`	One of `c("z","inverse","range","robust")`. Passed directly to `normalize_vec()`:
	`"z"` z-score (mean 0, sd 1).
	`"inverse"` Rank-based inverse-normal transform (Rankit; default).
	`"range"` Min-max scaling to `feature_range` (default `[0, 1]`).
	`"robust"` Median/MAD scaling.

skip_cols	Character vector of column names to leave untouched even if they are numeric (e.g. c("age", "BMI")). Ignored when cols is explicitly supplied.
...	Additional arguments forwarded to normalize_vec() (e.g. feature_range, invnorm_denominator, ties).

Value

The input data with the selected columns replaced by their normalised values. Class (tibble, data.frame, etc.) is preserved.

See Also

normalize_vec() for single-vector normalisation.

Examples

```
# Build a tiny data frame with pre-computed marker columns
df <- data.frame(
  age    = c(45, 52, 61),
  HOMA_IR = c(1.2, 3.4, 2.1),
  TyG     = c(8.1, 9.0, 8.6),
  NLR     = c(2.1, 3.5, 1.8)
)

marker_cols <- c("HOMA_IR", "TyG", "NLR")

# z-score normalise marker columns only
hm_normalize(df, cols = marker_cols, method = "z")

# Inverse-normal transform, leaving age untouched
hm_normalize(df, method = "inverse", skip_cols = "age")
```

---

hormone_markers *Compute a suite of hormone ratio markers with QA and verbose summaries*

Description

Ratios computed:

- FAI = (total_testosterone / SHBG) * 100
- LH_FSH = LH / FSH
- E2_P = estradiol / progesterone
- T3_T4 = free_T3 / free_T4
- ARR = aldosterone / renin
- Ins_Glu = insulin / glucagon
- GH_IGF1 = GH / IGF1
- PRL_T = prolactin / total_testosterone
- CAR_slope = (cortisol_30 - cortisol_0) / 30

Usage

```
hormone_markers(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error", "warn", "ignore"),
  na_warn_prop = 0.2,
  verbose = TRUE
)
```

Arguments

data	Data frame or tibble with mapped hormone inputs.
col_map	Named list mapping the required keys to column names: total_testosterone, SHBG, LH, FSH, estradiol, progesterone, free_T3, free_T4, aldosterone, renin, insulin, glucagon, GH, IGF1, prolactin, cortisol_0, cortisol_30.
na_action	One of c("keep","omit","error","warn","ignore"). "keep"/"ignore" leave NAs; "omit" drops rows with any NA in used inputs; "error" aborts; "warn" also warns about high missingness.
na_warn_prop	Proportion in $[0, 1]$ for high-missingness warnings when na_action = "warn". Default 0.2.
verbose	Logical; if TRUE (default), prints column mapping, input availability, inference notes, physiological range information (informational only, values not altered), computing markers, and a per-column results summary.

Details

Some inputs may be inferred when missing (for example, free_T3 from TSH + free_T4, or GH from IGF1) using internal heuristics. These inferred values are intended for exploratory feature engineering only and must not be treated as clinical substitutes for directly measured assays. Ratios such as FAI, ARR, and CAR_slope have established literature usage; several other outputs are simple arithmetic composites included for feature-engineering convenience and may not have a single canonical derivation paper.

Value

Tibble with one column per computable ratio. If an ID column is detected in data (e.g. id, IID, participant_id), it is prepended as the first output column.

References

Sowers MR, Zheng H, McConnell D, Nan B, Karvonen-Gutierrez CA, Randolph JF (2009). "Testosterone, sex hormone-binding globulin and free androgen index among adult women: chronological and ovarian aging." *Human Reproduction*, **24**(9), 2276–2285. doi:10.1093/humrep/dep209. Funder JW, Carey RM, Mantero F, Murad MH, Reincke M, Shibata H, Stowasser M, Young WF (2016). "The Management of Primary Aldosteronism: Case Detection, Diagnosis, and Treatment: An Endocrine Society Clinical Practice Guideline." *The Journal of Clinical Endocrinology & Metabolism*, **101**(5), 1889–1916. doi:10.1210/jc.20154061. Clow A, Thorn L, Evans P, Hucklebridge F (2004).

"The awakening cortisol response: methodological issues and significance." *Stress*, **7**(1), 29–37. doi:10.1080/10253890410001667205.

Examples

```
df <- data.frame(
  TT = c(15, 12), SHBG = c(40, 35), LH = c(5, 6), FSH = c(4, 5),
  E2 = c(100, 120), Prog = c(0.5, 0.6), fT3 = c(4.5, 4.2),
  fT4 = c(15, 14), Aldo = c(200, 180), Renin = c(10, 12),
  Ins = c(60, 70), Gluc = c(8, 9), GH = c(1.2, 1.0),
  IGF1 = c(180, 160), Prl = c(10, 12), Cort0 = c(400, 380),
  Cort30 = c(600, 580)
)
col_map <- list(
  total_testosterone = "TT", SHBG = "SHBG", LH = "LH", FSH = "FSH",
  estradiol = "E2", progesterone = "Prog", free_T3 = "fT3",
  free_T4 = "fT4", aldosterone = "Aldo", renin = "Renin",
  insulin = "Ins", glucagon = "Gluc", GH = "GH", IGF1 = "IGF1",
  prolactin = "Prl", cortisol_0 = "Cort0", cortisol_30 = "Cort30"
)
hormone_markers(df, col_map = col_map)
```

---

iAge *Compute a simplified Inflammatory Age Index (iAge) with QA and verbose summaries*

Description

Implements a linear proxy for immunosenescence based on key inflammatory biomarkers, conceptually inspired by the inflammatory aging clock (iAge) literature. This simplified iAge is computed as a weighted sum of C-reactive protein (CRP), interleukin-6 (IL6), and tumor necrosis factor-alpha (TNFa).

Usage

```
iAge(
  data,
  col_map = NULL,
  weights = c(CRP = 0.33, IL6 = 0.33, TNFa = 0.34),
  verbose = TRUE,
  na_action = c("keep", "omit", "error", "ignore", "warn"),
  na_warn_prop = 0.2
)
```

Arguments

`data`	A data.frame or tibble containing the biomarker columns mapped by `col_map`.
`col_map`	Named list mapping:

- CRP -> column name for C-reactive protein (mg/L)
- IL6 -> column name for interleukin-6 (pg/mL)
- TNFa -> column name for tumor necrosis factor-alpha (pg/mL)

`weights` Named numeric vector of weights for each marker (must sum to 1). Defaults to c(CRP = 0.33, IL6 = 0.33, TNFa = 0.34).

`verbose` Logical; if TRUE, prints column mapping and computing messages.

`na_action` One of c("keep","omit","error","ignore","warn") controlling how missing inputs affect iAge:

- "keep": return NA for rows where any required marker is NA (default; consistent with other functions).
- "omit": ignore NAs in the weighted sum (treat missing markers as 0).
- "error": abort if any required marker contains NA.
- "ignore": alias of "omit".
- "warn": alias of "omit" but emits missingness warnings (per na_warn_prop).

`na_warn_prop` Proportion in $[0, 1]$ above which a high-missingness warning is emitted when `na_action = "warn"`. Default 0.2.

Details

By default, rows with any missing required marker return NA in the index (consistent with the default behavior of other package functions). Optional diagnostics can warn on high missingness and implausible negative values. Verbose mode prints step-by-step progress and a final summary.

Assumed units (no automatic unit conversion):

- CRP: mg/L
- IL6: pg/mL
- TNFa: pg/mL

Note:

- The original iAge model in Sayed et al. (Nature Aging, 2021) is a multi-marker machine learning model. This function provides a simple, linear proxy using three canonical inflammatory biomarkers. It is not identical to the original published iAge but is inspired by its rationale.
- This proxy is intended for exploratory feature engineering and cohort-level analyses. It must not be treated as a validated replacement for the published iAge model or used as a standalone clinical decision metric.

Value

A tibble with one column:

- iAge (numeric): the computed inflammatory age index.

References

Sayed N, others (2021). "An inflammatory aging clock (iAge) predicts multimorbidity, immunosenescence, frailty and cardiovascular aging." *Nature Aging*, **1**, 598–610. doi:10.1038/s43587021-00082y. (conceptual background; not method-identical to this implementation) Harris TB, Ferrucci L, Tracy RP, Corti MC, Wacholder S, Ettinger WH, Heimovitz H, Cohen HJ, Wallace R (1999). "Associations of Elevated Interleukin-6 and C-Reactive Protein Levels with Mortality in the Elderly." *The American Journal of Medicine*, **106**(5), 506–512. doi:10.1016/S00029343(99)000662. Bruunsgaard H, Ladelund S, Pedersen AN, Schroll M, Jorgensen T, Pedersen BK (2003). "Predicting death from tumour necrosis factor-alpha and interleukin-6 in 80-year-old people." *Clinical and Experimental Immunology*, **132**(1), 24–31. doi:10.1046/j.13652249.2003.02137.x.

See Also

`impute_missing()`, `glycemic_markers()`

Examples

```
library(tibble)
df <- tibble(
  CRP  = c(1.2, 3.5, NA),  # mg/L
  IL6  = c(2.0, 4.1, 1.5), # pg/mL
  TNFa = c(1.0, 1.8, 0.9)  # pg/mL
)
# Default behavior (rows with any missing marker return NA)
iAge(
  df,
  col_map = list(CRP = "CRP", IL6 = "IL6", TNFa = "TNFa")
)

# Keep NA if any marker missing in a row
iAge(
  df,
  col_map = list(CRP = "CRP", IL6 = "IL6", TNFa = "TNFa"),
  na_action = "keep"
)

# Verbose output
iAge(
  df,
  col_map = list(CRP = "CRP", IL6 = "IL6", TNFa = "TNFa"),
  verbose = TRUE
)
```

`impute_mice` *Impute missing values via Multiple Imputation (mice)*

Description

Wraps mice to impute only numeric columns; non-numeric columns are untouched. Requires the mice package to be installed (Suggests). If no numeric columns contain NAs, `data` is returned unchanged.

Usage

```
impute_mice(data, m = 5, cols = NULL, verbose = FALSE, ...)
```

Arguments

`data`	A data.frame or tibble containing missing values.
`m`	Integer; number of imputations to run (passed to mice). Default 5.
`cols`	Optional character vector of numeric columns to impute. Defaults to all numeric columns with at least one NA.
`verbose`	Logical; if TRUE, prints progress and a completion summary. Default FALSE.
`...`	Additional arguments passed to mice::mice().

Details

Notes:

- At least two numeric columns are typically needed by mice to borrow strength.
- This function runs `m` imputations and returns the first completed dataset.
- Messages from mice are suppressed; use `verbose = TRUE` here for high-level progress.

Value

A data.frame/tibble with numeric columns imputed by mice.

References

Rubin DB (1987). *Multiple Imputation for Nonresponse in Surveys*. Wiley. doi:10.1002/9780470316696.
van Buuren S, Groothuis-Oudshoorn K (2011). "mice: Multivariate Imputation by Chained Equations in R." *Journal of Statistical Software*, **45**(3), 1–67. doi:10.18637/jss.v045.i03.

Examples

```
if (requireNamespace("mice", quietly = TRUE)) {
  df <- tibble::tibble(a = c(1, NA, 3), b = c(2, 4, NA), c = 5)
  impute_mice(df, m = 2, verbose = TRUE)
}
```

`impute_missforest` *Impute missing values via random forest (missForest)*

Description

Wraps missForest to impute numeric columns using non-parametric random forests. Requires the missForest package to be installed (Suggests). Non-numeric columns are untouched. If no numeric columns contain NAs, `data` is returned unchanged.

Usage

```
impute_missforest(data, ntree = 100, cols = NULL, verbose = FALSE, ...)
```

Arguments

`data`	A data.frame or tibble containing missing values.
`ntree`	Integer; number of trees to grow in each forest (passed to missForest). Default 100.
`cols`	Optional character vector of numeric columns to impute. Defaults to all numeric columns with at least one NA.
`verbose`	Logical; if TRUE, prints progress and a completion summary. Default FALSE.
`...`	Additional arguments passed to missForest::missForest().

Details

Notes:

- missForest uses iterative RF training; it can be slow on wide/high-NA data.
- Errors (e.g., insufficient unique values) are caught and a deterministic mean imputation fallback is applied to the selected numeric columns.

Value

A data.frame/tibble with numeric columns imputed by missForest (or mean fallback).

References

Stekhoven DJ, Buhlmann P (2012). "MissForest—non-parametric missing value imputation for mixed-type data." *Bioinformatics*, **28**(1), 112–118. doi:10.1093/bioinformatics/btr597.

Examples

```
if (requireNamespace("missForest", quietly = TRUE)) {
  df <- tibble::tibble(a = c(1, NA, 3), b = c(2, 4, NA), c = 5)
  impute_missforest(df, ntree = 50, verbose = TRUE)
}
```

`impute_missing` *Impute missing values in a data.frame or tibble (simple, column-wise)*

Description

Performs deterministic, per-column imputation for numeric variables:

- "mean": replace NAs with the column mean
- "median": replace NAs with the column median
- "zero": replace NAs with 0
- "constant": replace NAs with the single value given in `constant`

Usage

```
impute_missing(
  data,
  method = c("mean", "median", "zero", "constant"),
  cols = NULL,
  constant = NULL,
  na_warn_prop = 0.2,
  verbose = FALSE
)
```

Arguments

`data`	A data.frame or tibble containing missing values.
`method`	Character; one of c("mean","median","zero","constant").
`cols`	Optional character vector of column names to impute. Defaults to all numeric columns in `data` that contain at least one NA.
`constant`	Numeric; single value to use when `method = "constant"`.
`na_warn_prop`	Numeric in $[0, 1]$; threshold for high-missingness warnings per column. Default 0.2.
`verbose`	Logical; if TRUE, prints progress and a completion summary. Default FALSE.

Details

Non-numeric columns are left untouched. If `cols = NULL`, all numeric columns that have at least one NA are selected automatically. NA positions are the only values modified; non-NA entries are preserved as-is.

Quality checks:

- Warns for high-missingness columns (`>= na_warn_prop`).
- Warns and skips imputation when a column has no non-NA values (mean/median undefined).
- Coerces only numeric columns; non-numerics in `cols` are skipped with a warning.

Value

A data.frame/tibble of the same dimensions as `data`, with the specified columns' missing values imputed.

Examples

```
df <- tibble::tibble(a = c(1, NA, 3), b = c(NA, NA, 2), c = letters[1:3])
impute_missing(df, method = "mean")
impute_missing(df, method = "median", verbose = TRUE)
impute_missing(df, method = "constant", constant = -1, cols = "a")
```

---

inflammatory_markers *Compute inflammatory indices (classic, eosinophil, or both)*

Description

Panels:

- classic: NLR, PLR, LMR, dNLR, SII, SIRI, AISI, CRP_category
- eos: NLR, PLR, LMR, NER, SII, SIRI, PIV, CLR, CAR, PCR, mGPS, ESR (if mapped)
- both: union of classic and eos panels

Usage

```
inflammatory_markers(
  data,
  col_map = NULL,
  panel = c("auto", "classic", "eos", "both"),
  na_action = c("keep", "omit", "error"),
  verbose = TRUE
)
```

Arguments

`data`	data.frame or tibble
`col_map`	named list mapping keys to column names in `data`. Keys: neutrophils, lymphocytes, monocytes, platelets, WBC, CRP, albumin, eosinophils, ESR.
`panel`	one of c("auto","classic","eos","both"). "auto" uses presence of eosinophils key.
`na_action`	one of c("keep","omit","error"). Default "keep" propagates NA in outputs where inputs are missing.
`verbose`	logical; if `TRUE` (default), prints column mapping, optional input availability, physiological range information (informational only), the list of markers being computed, and a results summary.

Details

Derived markers:

- NLR = neutrophils / lymphocytes
- PLR = platelets / lymphocytes
- LMR = lymphocytes / monocytes
- dNLR = neutrophils / (WBC - neutrophils) when WBC available
- SII = platelets * neutrophils / lymphocytes
- SIRI = neutrophils * monocytes / lymphocytes
- AISI = neutrophils * monocytes * platelets / lymphocytes
- CRP_category: "low" (<1 mg/L), "moderate" (1-3 mg/L), "high" (>3 mg/L) when CRP available
- Eosinophil-panel extras: NER = neutrophils / eosinophils; PIV = platelets * neutrophils * monocytes / lymphocytes; CLR = CRP/lymphocytes; CAR = CRP/albumin; PCR = platelets/CRP; mGPS (CRP, albumin); ESR passthrough.

Note:

- These outputs are deterministic algebraic indices computed from the mapped laboratory variables. They are intended for feature engineering and descriptive analyses, not as standalone diagnosis/prognosis tools.
- References below document commonly used index definitions or interpretation conventions directly used in this implementation.

Value

tibble with selected inflammatory indices, with ID column prepended if detected (e.g. `id`, `IID`, `participant_id`).

References

Zahorec R (2001). "Ratio of neutrophil to lymphocyte counts–rapid and simple parameter of systemic inflammation and stress." *Bratislavske lekarske listy*, **102**(1), 5–14. No DOI identified; PMID: 11723675, https://pubmed.ncbi.nlm.nih.gov/11723675/. Hu B, others (2014). "Systemic Immune-Inflammation Index Predicts Prognosis of Patients after Curative Resection for Hepatocellular Carcinoma." *Clinical Cancer Research*, **20**(23), 6212–6222. doi:10.1158/1078-0432.CCR140442. Qi Q, others (2016). "A novel systemic inflammation response index (SIRI) for predicting the survival of patients with pancreatic cancer after chemotherapy." *Cancer*, **122**(14), 2158–2167. doi:10.1002/cncr.30057. Proctor MJ, others (2011). "An inflammation-based prognostic score (mGPS) predicts cancer survival independent of tumour site: a Glasgow Inflammation Outcome Study." *British Journal of Cancer*, **104**(4), 726–734. doi:10.1038/sj.bjc.6606087. Pearson TA, others (2003). "Markers of inflammation and cardiovascular disease: a statement for healthcare professionals from the CDC and AHA." *Circulation*, **107**(3), 499–511. doi:10.1161/01.CIR.0000052939.59093.45.

Examples

```
# Quick smoke-test
df <- data.frame(neutrophils = 4, lymphocytes = 2, monocytes = 0.5,
                 platelets = 200, WBC = 7, CRP = 2.5)
inflammatory_markers(df, panel = "classic", na_action = "keep", verbose = FALSE)


df <- data.frame(
  neutrophils = c(4, 2),
  lymphocytes = c(2, 0),
  monocytes   = c(0.5, 0.3),
  platelets   = c(200, 150),
  WBC         = c(7, 4.5),
  CRP         = c(2.5, 0.8),
  albumin     = c(40, 42),
  eosinophils = c(0.2, 0.1),
  ESR         = c(12, 15)
)
cm <- list(
  neutrophils = "neutrophils", lymphocytes = "lymphocytes", monocytes = "monocytes",
  platelets = "platelets", WBC = "WBC", CRP = "CRP", albumin = "albumin",
  eosinophils = "eosinophils", ESR = "ESR"
)
classic_cm <- cm; classic_cm$eosinophils <- NULL; classic_cm$ESR <- NULL
inflammatory_markers(df, classic_cm, panel = "classic", na_action = "keep")
inflammatory_markers(df, cm, panel = "eos", na_action = "keep", verbose = TRUE)
```

---

isi_score *Insomnia Severity Index (ISI) scoring*

Description

Insomnia Severity Index (ISI) scoring

Usage

```
isi_score(
  data,
  col_map = list(),
  na_action = c("keep", "omit", "error"),
  missing_prop_max = 0.2,
  impute = c("none", "mean"),
  prefix = "ISI",
  verbose = TRUE
)
```

Arguments

data	Data frame containing questionnaire item columns.
col_map	Named list mapping canonical item IDs to column names; defaults assume items are already named.
na_action	How to handle rows with missing items: keep, omit, or error.
missing_prop_max	Maximum allowed proportion of missing items per row before the score is set to NA.
impute	Imputation strategy for missing items when under the threshold: none or mean (row-wise mean).
prefix	Prefix for output column names.
verbose	Logical; if TRUE, emits informational messages about column resolution and scoring progress via hm_inform().

Value

A tibble of score columns only: ISI_total and ISI_severity (factor). Input columns are not included.

References

Bastien CH, Vallières A, Morin CM (2001). "Validation of the Insomnia Severity Index as an Outcome Measure for Insomnia Research." *Sleep Medicine*, **2**(4), 297–307. doi:10.1016/S1389-9457(00)000654.

Examples

```
df <- data.frame(isi_01 = 0, isi_02 = 1, isi_03 = 2, isi_04 = 1, isi_05 = 0, isi_06 = 1, isi_07 = 2)
isi_score(df)
```

---

k10_score *K10 scoring*

Description

K10 scoring

Usage

```
k10_score(
  data,
  col_map = list(),
  na_action = c("keep", "omit", "error"),
  missing_prop_max = 0.2,
  impute = c("none", "mean"),
  prefix = "K10",
  verbose = TRUE
)
```

Arguments

data	Data frame containing questionnaire item columns.
col_map	Named list mapping canonical item IDs to column names; defaults assume items are already named.
na_action	How to handle rows with missing items: keep, omit, or error.
missing_prop_max	Maximum allowed proportion of missing items per row before the score is set to NA.
impute	Imputation strategy for missing items when under the threshold: none or mean (row-wise mean).
prefix	Prefix for output column names.
verbose	Logical; if TRUE, emits informational messages about column resolution and scoring progress via hm_inform().

Value

A tibble of score columns only: K10_total. Input columns are not included.

Note

K10 items are summed as provided. The original scale uses 1–5 coding (total 10–50); some implementations subtract 1 (0–4, total 0–40). The function accepts either coding, as no reverse-scored items are involved and min_val/max_val only affect reversal.

References

Kessler RC, Andrews G, Colpe LJ, Hiripi E, Mroczek DK, Normand ST, Walters EE, Zaslavsky AM (2002). "Short screening scales to monitor population prevalences and trends in non-specific psychological distress." *Psychological Medicine*, **32**(6), 959–976. doi:10.1017/S0033291702006074.

Examples

```
df <- data.frame(k10_01 = 0, k10_02 = 1, k10_03 = 2, k10_04 = 1, k10_05 = 0,
                 k10_06 = 1, k10_07 = 2, k10_08 = 1, k10_09 = 0, k10_10 = 1)
k10_score(df)
```

---

k6_score *K6 scoring*

Description

K6 scoring

Usage

```
k6_score(
  data,
  col_map = list(),
  na_action = c("keep", "omit", "error"),
  missing_prop_max = 0.2,
  impute = c("none", "mean"),
  prefix = "K6",
  cutoff = 13,
  verbose = TRUE
)
```

Arguments

`data`	Data frame containing questionnaire item columns.
`col_map`	Named list mapping canonical item IDs to column names; defaults assume items are already named.
`na_action`	How to handle rows with missing items: `keep`, `omit`, or `error`.
`missing_prop_max`	Maximum allowed proportion of missing items per row before the score is set to `NA`.
`impute`	Imputation strategy for missing items when under the threshold: `none` or `mean` (row-wise mean).
`prefix`	Prefix for output column names.
`cutoff`	Threshold for the K6 case flag.
`verbose`	Logical; if `TRUE`, emits informational messages about column resolution and scoring progress via `hm_inform()`.

Value

A tibble of score columns only: `K6_total` and `K6_case`. Input columns are not included.

References

Kessler RC, Andrews G, Colpe LJ, Hiripi E, Mroczek DK, Normand ST, Walters EE, Zaslavsky AM (2002). "Short screening scales to monitor population prevalences and trends in non-specific psychological distress." *Psychological Medicine*, **32**(6), 959–976. doi:10.1017/S0033291702006074.
Prochaska JJ, Sung H, Max W, Shi Y, Ong M (2012). "Validity Study of the K6 Scale as a Measure of Moderate Mental Distress Based on Mental Health Treatment Need and Utilization." *International Journal of Methods in Psychiatric Research*, **21**(2), 88–97. doi:10.1002/mpr.1349. (validation study)

Examples

```
df <- data.frame(k6_01 = 0, k6_02 = 1, k6_03 = 2, k6_04 = 1, k6_05 = 0, k6_06 = 1)
k6_score(df)
```

`kidney_failure_risk` *Kidney Failure Risk Equation (KFRE, 2- and 5-year risk)*

Description

Compute 2- and 5-year risk of end-stage kidney disease using the original 4-variable KFRE (Tangri et al., 2011) with optional data-quality diagnostics, and verbose progress reporting.

Usage

```
kidney_failure_risk(
  data,
  col_map = list(age = "age", sex = "sex", eGFR = "eGFR", UACR = "UACR"),
  na_action = c("keep", "error", "omit", "warn"),
  na_warn_prop = 0.2,
  verbose = TRUE
)
```

Arguments

`data`	A data.frame or tibble containing at least the columns mapped in `col_map`.
`col_map`	Named list mapping: • `age` -> age in years • `sex` -> sex code (1 = male, 2 = female) • `eGFR` -> estimated GFR (mL/min/1.73 m^2) • `UACR` -> urine albumin-to-creatinine ratio (mg/g)
`na_action`	One of c("keep","error","omit","warn"). Default "keep" to preserve previous behavior: • "keep": propagate NA/NaN through logs and outputs. • "error": abort if any required input contains missing values. • "omit": drop rows with NA in required inputs before computation. • "warn": like "keep" but emits high-missingness warnings.
`na_warn_prop`	Numeric in $[0, 1]$; per-variable threshold for high-missingness warnings. Default 0.2.
`verbose`	Logical; if TRUE, prints stepwise messages and a completion summary. Default TRUE.

Details

This function preserves prior behavior by default:

- Inputs are taken as-is; NA values propagate to outputs (na_action = "keep").
- No capping or out-of-range checks are applied.

Units (no automatic conversion):

- age: years; sex: 1 = male, 2 = female
- eGFR: mL/min/1.73 m^2
- UACR: mg/g (albumin-to-creatinine ratio)

Details

- Prognostic index: `PI = 0.220 x log(age) + (-0.556) x log(eGFR) + 0.451 x log(UACR) + 0.391 x (male)` where male = 1 if sex == 1, else 0.
- Baseline survival: S0(2y) = 0.934, S0(5y) = 0.881 (Tangri 2011).
- Risks: KFRE_t = 1 - (S0_t ^ exp(PI)).
- The 2016 JAMA study provides a large, multinational validation of the KFRE in humans.
- This implementation computes the original 4-variable linear predictor and does not apply recalibration or alternative coefficient sets.

Value

A tibble with:

- `KFRE_2yr` risk (0-1) at 2 years
- `KFRE_5yr` risk (0-1) at 5 years

References

Tangri N, Stevens LA, Griffith J, others (2011). "A predictive model for progression of chronic kidney disease to kidney failure." *JAMA*, **305**(15), 1553–1559. doi:10.1001/jama.2011.451. Tangri N, Grams ME, Levey AS, others (2016). "Multinational assessment of accuracy of equations for predicting risk of kidney failure: a meta-analysis." *JAMA*, **315**(2), 164–174. doi:10.1001/jama.2015.18202.

See Also

`inflammatory_markers()`, `iAge()`, `impute_missing()`

Examples

```
library(tibble)
df <- tibble(
  age  = c(65, 72),
  sex  = c(1, 2),          # 1 = male, 2 = female
  eGFR = c(45, 22),        # mL/min/1.73 m^2
  UACR = c(300, 1200)      # mg/g
)
# Default behavior (NA propagate, no extreme checks)
kidney_failure_risk(
  data = df,
  col_map = list(age = "age", sex = "sex", eGFR = "eGFR", UACR = "UACR")
)

# With verbose output
```

```
kidney_failure_risk(
  data = df,
  col_map = list(age = "age", sex = "sex", eGFR = "eGFR", UACR = "UACR"),
  verbose = TRUE
)
```

---

kyn_trp_ratio *Kynurenine/Tryptophan Ratio (KTR)*

Description

Computes the ratio of kynurenine to tryptophan, a marker of IDO activity and immune activation.

Usage

```
kyn_trp_ratio(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error", "ignore", "warn"),
  verbose = TRUE
)
```

Arguments

data	A data.frame or tibble with kynurenine and tryptophan concentrations.
col_map	Named list with: • kynurenine: column for kynurenine (nmol/L) • tryptophan: column for tryptophan (mumol/L)
na_action	One of c("keep","omit","error","ignore","warn").
verbose	Logical; if TRUE (default), prints column mapping and a per-column results summary.

Details

KTR is calculated as Kyn (nmol/L) divided by Trp (mumol/L). Elevated KTR indicates increased tryptophan catabolism via the kynurenine pathway, often reflecting inflammation and cell-mediated immune activation.

Inputs should already be in Kyn (nmol/L) and Trp (mumol/L).

Value

A tibble with one column: kyn_trp_ratio (numeric). If an ID column is detected, it is prepended.

References

Fuchs D, Moller AA, Reibnegger G, Werner ER, Werner-Felmayer G, Dierich MP, Wachter H (1998). "Serum kynurenine-to-tryptophan ratio increases with disease progression in HIV-1 infection." *Clinical Chemistry*, **44**(4), 858–862. doi:10.1093/clinchem/44.4.858, PMID:9555676.; Damerell V, Midttun O, Ulvik A, et al. (2025). "Circulating tryptophan-kynurenine pathway metabolites are associated with all-cause mortality among patients with stage I–III colorectal cancer." *International Journal of Cancer*, **156**(3), 552–565. doi:10.1002/ijc.35183. (clinical application in colorectal cancer)

Examples

```
# columns named exactly as the required keys (auto-detected)
df <- data.frame(kynurenine = c(2500, 3100, 2700), tryptophan = c(55, 48, 62))
kyn_trp_ratio(df, verbose = FALSE)

# non-standard column names require explicit col_map
df2 <- data.frame(Kyn_nM = c(2500, 3100, 2700), Trp_uM = c(55, 48, 62))
kyn_trp_ratio(df2, col_map = list(kynurenine = "Kyn_nM", tryptophan = "Trp_uM"),
              verbose = FALSE)
```

---

lipid_markers *Calculate lipid-panel markers, Visceral Adiposity Index (VAI), Lipid Accumulation Product (LAP), and TyG-BMI index*

Description

Given total cholesterol, HDL, TG (and optionally LDL, ApoB/ApoA1, waist, BMI, glucose), computes:

- `non_HDL_c`, `remnant_c`
- `ratio_TC_HDL`, `ratio_TG_HDL`, `ratio_LDL_HDL`
- `ApoB_ApoA1`
- `VAI_Men`, `VAI_Women`
- `LAP_Men`, `LAP_Women`
- `TyG_BMI`

Usage

```
lipid_markers(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error", "ignore", "warn"),
  na_warn_prop = 0.2,
  verbose = TRUE
)
```

Arguments

data	A data.frame or tibble containing your lipid (and optional anthropometry/glucose) data.
col_map	Named list mapping: • TC -> total cholesterol • HDL_c -> HDL-C • TG -> triglycerides • LDL_c -> (optional) LDL-C; if absent, estimated via Friedewald • ApoB, ApoA1 -> (optional) apolipoproteins • waist -> (optional) waist circumference (cm) • BMI -> (optional) body mass index (kg/m^2) • glucose -> (optional) fasting glucose (mmol/L); used for TyG_BMI
na_action	One of c("keep","omit","error","ignore","warn"). • keep/ignore: compute and propagate NA in outputs • omit: drop rows with NA in required inputs (TC, HDL_c, TG) • error: abort if any required input contains NA • warn: like keep, but emit missingness warnings
na_warn_prop	Proportion (0-1) threshold for high-missingness warnings when na_action = "warn". Default 0.2.
verbose	Logical; if TRUE (default), prints step-by-step progress including column mapping, optional input availability, pre-computation notes, physiological range information (informational only, values are not altered), the list of markers being computed with their inputs, and a per-column results summary.

Details

Assumed units (no automatic conversion except where noted):

- TC, HDL_c, TG, LDL_c: mmol/L
- glucose: mmol/L (converted to mg/dL internally for TyG_BMI)
- waist: cm
- BMI: kg/m^2

Pre-computation (one level deep):

- If BMI is absent but weight (kg) and height (m or cm) are present, BMI is computed automatically.
- If glucose is absent but G0 is present (or vice versa), the alias is derived automatically.
- If LDL_c is absent, it is always estimated via Friedewald (TC - HDL - TG/2.2, mmol/L form). An informational message is emitted when verbose = TRUE.

Value

A tibble with computed lipid markers. Required outputs (always present): `non_HDL_c`, `remnant_c`, `ratio_TC_HDL`, `ratio_TG_HDL`, `ratio_LDL_HDL`, `ApoB_ApoA1`. Optional outputs (present when inputs available): `VAI_Men`, `VAI_Women` (waist + BMI required); `LAP_Men`, `LAP_Women` (waist required); `TyG_BMI` (BMI + glucose required). If an ID column is detected in `data` (e.g. `id`, `IID`, `participant_id`), it is prepended as the first output column.

References

Friedewald WT, Levy RI, Fredrickson DS (1972). "Estimation of the concentration of LDL cholesterol in plasma, without use of preparative ultracentrifuge." *Clinical Chemistry*, **18**(6), 499–502. doi:10.1093/clinchem/18.6.499. Amato MC, Giordano C, Galia M, Criscimanna A, Vitabile S, Midiri M, Galluzzo A (2010). "Visceral Adiposity Index: A Reliable Indicator of Visceral Fat Function Associated with Cardiometabolic Risk." *Diabetes Care*, **33**(4), 920–922. doi:10.2337/dc091825. Kahn HS (2005). "The Lipid Accumulation Product Performs Better than Body Mass Index as an Indicator of Cardiovascular Risk in Women." *BMC Cardiovascular Disorders*, **5**(1), 26. doi:10.1186/14712261526. Er L, Wu S, Chou H, Hsu L, Teng M, Sun Y, Ko Y (2016). "Triglyceride Glucose-Body Mass Index Is a Simple and Clinically Useful Surrogate Marker for Insulin Resistance in Nondiabetic Individuals." *PLOS ONE*, **11**(3), e0149731. doi:10.1371/journal.pone.0149731. Khamseh ME, Malek M, Abbasi R, Taheri E, others (2021). "Triglyceride Glucose Index and Related Parameters (Triglyceride Glucose-Body Mass Index and Triglyceride Glucose-Waist Circumference) Identify Nonalcoholic Fatty Liver and Liver Fibrosis in Individuals with Overweight/Obesity." *Metabolic Syndrome and Related Disorders*, **19**(3), 167–173. doi:10.1089/met.2020.0109. (clinical application)

Examples

```
df <- data.frame(TC = c(5.2, 6.1), HDL_c = c(1.3, 1.1), TG = c(1.8, 2.3),
                 LDL_c = c(3.2, 4.1), waist = c(88, 95), BMI = c(26, 29))
# Full verbose output (default)
lipid_markers(df)
# Suppress messaging for batch use
lipid_markers(df, verbose = FALSE)
# Pre-compute BMI from weight and height
df2 <- data.frame(TC = 5.2, HDL_c = 1.3, TG = 1.8, weight = 70, height = 175)
lipid_markers(df2, verbose = FALSE)
```

---

liver_fat_markers *Liver fat surrogates: HSI and NAFLD Liver Fat Score*

Description

Computes:

- HSI = 8 * (ALT/AST) + BMI + 2 (if female) + 2 (if diabetes)
- NAFLD-LFS = -2.89 + 1.18*MetS* + *0.45*Type2DM + 0.15*Insulin_u* + *0.04*AST - 0.94*(AST/ALT)

Usage

```
liver_fat_markers(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error", "ignore", "warn"),
  na_warn_prop = 0.2,
  verbose = TRUE
)
```

Arguments

`data`	Data frame with needed columns (see col_map).
`col_map`	Named list mapping: • Required for HSI: ALT, AST, BMI • Optional direct inputs: sex, diabetes, MetS, insulin • Optional to derive MetS or insulin: I0, waist, TG, HDL_c, sbp, bp_sys, bp_treated, glucose, G0
`na_action`	One of c("keep","omit","error","ignore","warn").
`na_warn_prop`	Proportion in $[0, 1]$ for high-missingness warnings when na_action = "warn". Default 0.2.
`verbose`	Logical; if TRUE, prints column mapping and computing messages.

Details

Assumptions/units:

- ALT, AST in U/L; BMI in kg/m^2.
- `insulin` is expected in muU/mL; if unavailable and `I0` is provided, `I0` is interpreted in pmol/L and converted to muU/mL via /6.
- MetS is taken directly if provided; otherwise derived using a simplified NCEP-ATP III style rule when sufficient inputs exist.
- Type2DM is taken from `diabetes` (logical or 0/1).

These scores are surrogate indices for research/feature-engineering use and are not validated as standalone clinical diagnostic tools.

Value

A tibble with columns HSI and NAFLD_LFS.

References

Lee J, Kim D, Kim HJ, Lee CH, Yang JI, Kim W, Kim YJ, Yoon J, Cho S, Sung M, Lee H (2010). "Hepatic steatosis index: a simple screening tool reflecting nonalcoholic fatty liver disease." *Digestive and Liver Disease*, **42**(7), 503–508. doi:10.1016/j.dld.2009.08.002. Kotronen A, Peltonen M, Hakkarainen A, Sevastianova K, Bergholm R, Johansson LM, Lundbom N, Rissanen A, Ridderstrale M, Groop L, Orho-Melander M, Yki-Järvinen H (2009). "Prediction of non-alcoholic fatty

liver disease and liver fat using metabolic and genetic factors." *Gastroenterology*, **137**(3), 865–872. doi:10.1053/j.gastro.2009.06.005.

Examples

```
df <- data.frame(ALT=20, AST=25, BMI=27, sex="female", diabetes=FALSE, I0=60)
liver_fat_markers(
  df,
  col_map = list(ALT="ALT", AST="AST", BMI="BMI",
                 sex="sex", diabetes="diabetes", I0="I0")
)
```

`liver_markers` *Compute liver-related indices (FLI, NFS, APRI, FIB-4, BARD, ALBI, MELD-XI) with validation and diagnostics*

Description

Given routine labs and anthropometry, computes:

- FLI - Fatty Liver Index (Bedogni et al. 2006)
- NFS - NAFLD Fibrosis Score (Angulo et al. 2007)
- APRI - AST-to-Platelet Ratio Index
- FIB4 - Fibrosis-4 Index
- BARD - BMI-AST/ALT-Diabetes score
- ALBI - Albumin-Bilirubin score
- MELD_XI - MELD excluding INR

Usage

```
liver_markers(
  data,
  col_map = NULL,
  verbose = TRUE,
  na_action = c("keep", "omit", "error"),
  na_warn_prop = 0.2
)
```

Arguments

`data`	A data.frame or tibble containing your liver and anthropometry data.
`col_map`	Named list mapping these keys -> column names in `data`: • `BMI` (kg/m^2), `waist` (cm), `TG` (mg/dL), `GGT` (U/L), • `age` (years), `AST` (U/L), `ALT` (U/L), `platelets` (10^9/L), • `albumin` (g/L), `diabetes` (0/1 or logical),

- `bilirubin` (mg/dL), `creatinine` (mg/dL).

`verbose` Logical; if `TRUE` (default), prints column mapping, input availability, physiological range information (informational only, values not altered), the list of markers being computed with their inputs, and a per-column results summary.

`na_action` One of `c("keep","omit","error")` controlling missing-data policy. Default "keep".

`na_warn_prop` Numeric in $[0, 1]$; per-variable threshold for high-missingness warnings. Default 0.2.

Details

Enhancements:

- Robust input validation (columns present, types) with informative errors.
- Configurable NA policy and optional extreme-value scanning/capping.
- Data-quality warnings (high missingness, non-positive logs, zero denominators).
- Verbose stepwise progress and completion summary.

Units (no automatic conversion):

- BMI: kg/m^2; Waist: cm; TG: mg/dL; GGT/AST/ALT: U/L; Platelets: 10^9/L; Albumin: g/L; Bilirubin: mg/dL; Creatinine: mg/dL.
- ALBI uses bilirubin in mumol/L internally (converted as bilirubin (mg/dL) * 17.1).

Formulas

- FLI = logistic(0.953*ln(TG)* + *0.139*BMI + 0.718*ln(GGT)* + *0.053*waist - 15.745) * 100
- NFS = -1.675 + 0.037*age* + *0.094*BMI + 1.13*diabetes* + *0.99*(AST/ALT) - 0.013*platelets* - *0.066*albumin (albumin in g/L; Angulo 2007 published coefficient -0.66 was for g/dL, divided by 10 here)
- APRI = (AST / 40) / platelets * 100; assumes AST upper limit of normal = 40 U/L
- FIB-4 = (age * AST) / (platelets * sqrt(ALT))
- BARD = +1 if BMI>=28, +2 if AST/ALT>=0.8, +1 if diabetes present; sum in 0-4
- ALBI = 0.66*log10(bilirubin (mumol/L))* - *0.0852*albumin (g/L)
- MELD-XI = 5.11*ln(bilirubin (mg/dL))* + *11.76*ln(creatinine (mg/dL)) + 9.44

Value

A tibble with one column per marker: `FLI`, `NFS`, `APRI`, `FIB4`, `BARD`, `ALBI`, `MELD_XI`. If an ID column is detected in `data` (e.g. `id`, `IID`, `participant_id`), it is prepended as the first output column.

References

Bedogni G, Bellentani S, Miglioli L, others (2006). "The Fatty Liver Index: a simple and accurate predictor of hepatic steatosis in the general population." *BMC Gastroenterology*, **6**, 33. doi:10.1186/1471230X633. Angulo P, Hui JM, Marchesini G, others (2007). "The NAFLD fibrosis score: a noninvasive system that identifies liver fibrosis in patients with NAFLD." *Hepatology*, **45**(4), 846–854. doi:10.1002/hep.21496. Wai CT, Greenson JK, Fontana RJ, others (2003). "A simple noninvasive index can predict both significant fibrosis and cirrhosis in patients with chronic hepatitis C." *Hepatology*, **38**(2), 518–526. doi:10.1053/jhep.2003.50346. Sterling RK, Lissen E, Clumeck N, others (2006). "Development of a simple noninvasive index to predict significant fibrosis in patients with HIV/HCV coinfection (FIB-4)." *Hepatology*, **43**(6), 1317–1325. doi:10.1002/hep.21178. Harrison SA, Oliver D, Arnold HL, others (2008). "Development and validation of a simple NAFLD clinical scoring system for identifying patients without advanced disease." *Gut*, **57**(10), 1441–1447. doi:10.1136/gut.2007.146019. Johnson PJ, Berhane S, Kagebayashi C, others (2015). "Assessment of liver function in patients with hepatocellular carcinoma: the ALBI grade." *Journal of Clinical Oncology*, **33**(6), 550–558. doi:10.1200/JCO.2014.57.9151. Heuman DM, Abou-Assi SG, Habib A, others (2006). "MELD-XI: A rational approach to "sickest first" liver transplantation in cirrhotic patients requiring anticoagulant therapy." *Liver Transplantation*, **13**(1), 30–37. doi:10.1002/lt.20906.

See Also

`inflammatory_markers()`, `kidney_failure_risk()`, `iAge()`

Examples

```
# Quick smoke-test
df <- data.frame(ALT = 25, AST = 20, BMI = 24, platelets = 250)
liver_markers(df, verbose = FALSE)


library(tibble)
df <- tibble(
  BMI           = 24,
  waist         = 80,
  TG = 150,
  GGT           = 30,
  age           = 45,
  AST           = 25,
  ALT           = 20,
  platelets     = 250,
  albumin       = 42,
  diabetes      = FALSE,
  bilirubin     = 1.0,
  creatinine    = 0.9
)
liver_markers(df)
liver_markers(df, verbose = FALSE)
```

marker_summary *Summarize marker outputs*

Description

Computes simple summaries (mean, SD, IQR) for numeric columns.

Usage

```
marker_summary(x, verbose = FALSE)
```

Arguments

x	Data frame returned by marker functions
verbose	Logical; if TRUE, prints progress messages

Value

Tibble with columns: variable, mean, sd, iqr

Examples

```
df <- data.frame(glucose = c(5.5, 6.1, 4.9), insulin = c(60, 88, 55),
                 bmi = c(24, 27, 22))
marker_summary(df)
```

mdq_score *Mood Disorder Questionnaire scoring*

Description

Mood Disorder Questionnaire scoring

Usage

```
mdq_score(
  data,
  col_map = list(),
  na_action = c("keep", "omit", "error"),
  missing_prop_max = 0.2,
  prefix = "MDQ",
  symptom_cutoff = 7,
  require_clustering = TRUE,
  require_impairment = TRUE,
  verbose = TRUE
)
```

Arguments

`data`	Data frame containing questionnaire item columns.
`col_map`	Named list mapping canonical item IDs to column names; defaults assume items are already named.
`na_action`	How to handle rows with missing items: `keep`, `omit`, or `error`.
`missing_prop_max`	Maximum allowed proportion of missing items per row before the score is set to `NA`.
`prefix`	Prefix for output column names.
`symptom_cutoff`	Minimum symptom count for a positive screen.
`require_clustering`	Require clustering item == 1 to be positive.
`require_impairment`	Require impairment item == 1 to be positive.
`verbose`	Logical; if `TRUE`, emits informational messages about column resolution and scoring progress via `hm_inform()`.

Value

A tibble of score columns only: `MDQ_symptom_count`, `MDQ_clustering`, `MDQ_impairment`, `MDQ_positive_screen`. Input columns are not included.

Note

The impairment column is expected to be **binary** (1 = impaired, 0 = not). The original MDQ impairment question uses a 4-category scale (1 = no problem, 2 = minor, 3 = moderate, 4 = serious); if the raw 4-category response is passed, the caller must recode it to binary (e.g., `impair_binary = as.integer(impair_raw >= 3)`) before scoring.

References

Hirschfeld RMA, Williams JBW, Spitzer RL, Calabrese JR, Flynn L, Keck PE, Lewis L, McElroy SL, Post RM, Rapport DJ, Russell JM, Sachs GS, Zajecka J (2000). "Development and Validation of a Screening Instrument for Bipolar Spectrum Disorder: The Mood Disorder Questionnaire." *American Journal of Psychiatry*, **157**(11), 1873–1875. doi:10.1176/appi.ajp.157.11.1873.

Examples

```
df <- data.frame(matrix(0, nrow = 1, ncol = 13))
names(df) <- sprintf("mdq_%02d", 1:13)
df$mdq_cluster <- 1; df$mdq_impair <- 1
mdq_score(df)
```

`metabolic_markers` *Aggregate selected metabolic marker groups*

Description

Aggregate selected metabolic marker groups

Usage

```
metabolic_markers(
  data,
  col_map = NULL,
 which = c("insulin", "adiposity_sds", "cardio", "lipid", "liver", "glycemic", "mets"),
  normalize = c("none", "z", "inverse", "range", "robust"),
  mode = c("both", "IS", "IR"),
  verbose = TRUE,
  na_action = c("keep", "omit", "error")
)
```

Arguments

`data`	A data.frame or tibble.
`col_map`	Named list for column mapping forwarded to underlying functions.
`which`	Character vector of groups to compute: c("insulin","adiposity_sds","cardio","lipid","liver","glycemic","m
`normalize`	One of c("none","z","inverse","range","robust").
`mode`	One of c("both","IS","IR").
`verbose`	Logical.
`na_action`	One of c("keep","omit","error"); forwarded to underlying calculators (HM-CS v2).

Value

Data frame with original columns plus derived markers.

Note

For references supporting liver, lipid, glycemic, MetS, adiposity and other domain-specific indices, see each underlying function's documentation (e.g. ?liver_markers, ?lipid_markers, ?glycemic_markers, ?metss, ?adiposity_sds). This wrapper omits repeated reference listings to avoid redundancy.

Aggregator wrapper. See underlying function help pages for full references: all_insulin_indices(), lipid_markers(), liver_markers(), glycemic_markers(), metss().

Examples

```
df <- data.frame(
  TC = 200, HDL_c = 50, TG = 150, LDL_c = 120,
  ALT = 30, AST = 20, BMI = 25
)
metabolic_markers(df, col_map = list(), which = c("lipid","liver"),
                  normalize = "none", mode = "both", verbose = FALSE, na_action = "keep")
```

---

metabolic_risk_features

Calculate metabolic risk feature flags (pediatric-friendly thresholds)

Description

Compute four binary risk flags from routine clinical measures:

- dyslipidemia
- insulin_resistance
- hyperglycemia (prediabetes-range glycemia)
- hypertension (BP >=95th percentile via z > 1.64)

Usage

```
metabolic_risk_features(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error", "ignore", "warn"),
  na_warn_prop = 0.2,
  verbose = TRUE
)
```

Arguments

data	A data.frame or tibble containing at least these numeric columns: • chol_total, chol_ldl, chol_hdl, triglycerides (mmol/L) • age_year (years) • z_HOMA (standardized HOMA-IR) • glucose (mmol/L) • HbA1c (mmol/mol; IFCC units) • bp_sys_z, bp_dia_z (BP z-scores)
col_map	Optional named list to map required keys to column names in data. Keys: c("chol_total","chol_ldl","chol_hdl","triglycerides","age_year", "z_HOMA","glucose","HbA1c","bp_sys Default NULL (use same names).

na_action	One of c("keep","omit","error","ignore","warn") controlling missing-data policy. • "keep": keep NA; outputs become NA where inputs are NA. • "omit": drop rows with NA in any required input. • "error": abort if any required input contains NA. • "ignore"/"warn": aliases of "keep"; "warn" also emits missingness diagnostics.
na_warn_prop	Numeric in $[0, 1]$; per-variable threshold for high-missingness warnings. Default 0.2.
verbose	Logical; if TRUE, prints column mapping and computing messages.

Details

By default, behavior matches prior implementation: required columns are validated, NA values are kept (propagate to outputs), no extreme-value checks or capping are applied, and a tibble with 0/1 factor flags is returned.

Units and criteria (no automatic unit conversion):

- Lipids (mmol/L): total cholesterol > 5.2 OR LDL-C > 3.4 OR HDL-C < 1.0 OR triglycerides > 1.1 (age 0-9) OR > 1.5 (age 10-19) => dyslipidemia = 1. Note: no TG cutoff is applied for adults aged >= 20 years.
- Insulin resistance: z_HOMA > 1.28 (~=90th percentile) => insulin_resistance = 1. z_HOMA is a within-sample or external z-score of HOMA-IR (see Matthews et al. 1985).
- Hyperglycemia: fasting glucose in (5.6, 6.9) mmol/L OR HbA1c in (39, 47) mmol/mol => hyperglycemia = 1. Boundaries are EXCLUSIVE (open intervals); boundary values (exactly 5.6 or 6.9 mmol/L; exactly 39 or 47 mmol/mol) are not flagged. ADA criteria use inclusive lower bound (>= 5.6 mmol/L, >= 39 mmol/mol).
- Hypertension: either BP z-score > 1.64 (~=95th percentile) for systolic or diastolic => hypertension = 1.

Value

A tibble with four factor columns (levels c("0","1")):

- dyslipidemia
- insulin_resistance
- hyperglycemia
- hypertension

Note

These flags are heuristic screening rules derived from published clinical guidelines. They are not validated diagnostic criteria and should not replace clinical judgment. The dyslipidemia and hypertension thresholds are designed for pediatric populations (ages 0-19); for adults >= 20, only TC, LDL-C, and HDL-C criteria contribute to the dyslipidemia flag.

References

National Heart, Lung, and Blood Institute (2011). "Expert Panel on Integrated Guidelines for Cardiovascular Health and Risk Reduction in Children and Adolescents: Summary Report." *Pediatrics*, **128**(Suppl 5), S213–S256. doi:10.1542/peds.20092107C. American Diabetes Association Professional Practice Committee (2024). "2. Diagnosis and Classification of Diabetes: Standards of Care in Diabetes—2024." *Diabetes Care*, **47**(Suppl 1), S20–S42. doi:10.2337/dc24S002. Flynn JT, Kaelber DC, Baker-Smith CM, Blowey D, Carroll AE, Daniels SR, de Ferranti SD, Dionne JM, Falkner B, Flinn SK, Gidding SS, Goodwin C, Leu MG, Powers ME, Rea C, Samuels J, Simasek M, Tran VT, Urbina EM (2017). "Clinical Practice Guideline for Screening and Management of High Blood Pressure in Children and Adolescents." *Pediatrics*, **140**(3), e20171904. doi:10.1542/peds.20171904. Matthews DR, Hosker JP, Rudenski AS, Naylor BA, Treacher DF, Turner RC (1985). "Homeostasis Model Assessment: Insulin Resistance and Beta-Cell Function from Fasting Plasma Glucose and Insulin Concentrations in Man." *Diabetologia*, **28**(7), 412–419. doi:10.1007/BF00280883.

See Also

`liver_markers()`, `lipid_markers()`, `kidney_failure_risk()`, `inflammatory_markers()`

Examples

```
df <- data.frame(
  chol_total = c(5.2, 6.4), chol_ldl = c(3.2, 4.1), chol_hdl = c(1.3, 1.0),
  triglycerides = c(1.8, 2.5), age_year = c(45, 60), z_HOMA = c(0.5, 1.2),
  glucose = c(5.5, 6.8), HbA1c = c(38, 46), bp_sys_z = c(0.2, 1.1),
  bp_dia_z = c(0.1, 0.9)
)
metabolic_risk_features(df)
```

---

metss *Metabolic Syndrome Severity Score (MetSSS)*

Description

Computes a continuous metabolic syndrome severity z-score using sex- and race-specific standardized components and coefficients (factor-loading style).

Behavior note:

- Parameters are selected per-row based on each row's (race, sex) key.
- All unique keys present in the data must have a matching entry in `params`.

Required columns (no unit conversion performed):

- waist (cm), bp_sys (mmHg), bp_dia (mmHg)
- TG, HDL_c, glucose (mmol/L)
- sex (1=male, 2=female)
- race (one of "NHW","NHB","HW", or accepted synonyms; "HA" is recognised by the normaliser but has no default params — see `params` argument)

Usage

```
metss(
  data,
 params = list(NHW_M = list(intercept = -2.344, waist = c(mean = 94, sd = 12.4, coef =
  0.846), TG = c(mean = 1.5, sd = 0.6, coef = 0.701), HDL = c(mean = 1.1, sd = 0.3,
  coef = -0.663), glucose = c(mean = 5.3, sd = 0.6, coef = 0.658), MAP = c(mean = 97,
  sd = 11, coef = 0.466)), NHW_F = list(intercept = -2.381, waist = c(mean = 89.7, sd =
  14.8, coef = 0.817), TG = c(mean = 1.28, sd = 0.91, coef = 0.679), HDL = c(mean =
  1.5, sd = 0.4, coef = -0.727), glucose = c(mean = 5.08, sd = 0.52, coef = 0.622), MAP
   = c(mean = 91,
    sd = 11, coef = 0.557)), NHB_M = list(intercept = -2.399, waist
  = c(mean = 92.8, sd = 13.1, coef = 0.83), TG = c(mean = 1.18, sd = 0.75, coef =
  0.551), HDL = c(mean = 1.27, sd = 0.37, coef = -0.598), glucose = c(mean = 5.55, sd =
   0.85, coef = 0.702), MAP = c(mean = 98, sd = 13, coef = 0.564)), NHB_F =
  list(intercept = -2.395, waist = c(mean = 96.4, sd = 16.4, coef = 0.858), TG = c(mean
  = 1.14, sd = 0.7, coef = 0.57), HDL = c(mean = 1.36, sd = 0.39, coef = -0.634),
   glucose = c(mean = 5.42, sd = 0.84,
    coef = 0.687), MAP = c(mean = 95, sd = 13,
  coef = 0.577)), HW_M = list(intercept = -2.377, waist = c(mean = 98.5, sd = 11.5,
  coef = 0.864), TG = c(mean = 1.95, sd = 1.19, coef = 0.724), HDL = c(mean = 1.13, sd
  = 0.3, coef = -0.62), glucose = c(mean = 5.67, sd = 0.9, coef = 0.624), MAP = c(mean
  = 97, sd = 11, coef = 0.448)), HW_F = list(intercept = -2.388, waist = c(mean = 97.9,
  sd = 14.2, coef = 0.858), TG = c(mean = 1.66, sd = 1.06, coef = 0.715), HDL = c(mean
   = 1.29, sd = 0.35, coef = -0.657),
    glucose = c(mean = 5.53, sd = 0.87, coef =
   0.644), MAP = c(mean = 91, sd = 11, coef = 0.512))),
  col_map = NULL,
  verbose = TRUE,
  na_action = c("keep", "omit", "error", "ignore", "warn"),
  na_warn_prop = 0.2,
  diagnostics = TRUE
)
```

Arguments

data	data.frame / tibble.
params	Named list keyed by "RACE_SEX" (e.g. "NHW_M"). Each element: list(intercept, waist, TG, HDL, glucose, MAP) where each component (except intercept) is a named numeric vector c(mean=, sd=, coef=). Default parameters are provided for NHW, NHB, and HW (male and female); no default HA parameters are included because Gurka et al. (2014) did not publish HA-specific coefficients. Passing race = "HA" with default params will raise an error; supply custom params if needed.
col_map	Optional named list mapping canonical keys (waist, TG, HDL, glucose, MAP) to actual column names in data. If NULL, column names are inferred automatically.
verbose	Logical; if TRUE, prints column mapping and computing messages.

na_action	One of c("keep","omit","error","ignore","warn") for required-input NAs. Default "keep".
na_warn_prop	Proportion (0-1) above which high-missingness warning fires when na_action='warn'. Default 0.2.
diagnostics	Logical; if TRUE (default) emit value/range diagnostic warnings (negative, out-of-range checks).

Details

Calculate Metabolic Syndrome Severity Score (MetSSS)

Value

tibble with one numeric column: MetSSS

References

Gurka MJ, Lilly CL, Oliver MN, DeBoer MD (2014). "An examination of sex and racial/ethnic differences in the metabolic syndrome among adults: A confirmatory factor analysis and a resulting continuous severity score." *Metabolism*, **63**(2), 218–225. doi:10.1016/j.metabol.2013.10.006. DeBoer MD, Gurka MJ, Woo JG, Morrison JA (2015). "Severity of metabolic syndrome and its association with risk for type 2 diabetes and cardiovascular disease." *Diabetologia*, **58**(12), 2745–2752. doi:10.1007/s0012501537595. (clinical application) DeBoer MD, Filipp SL, Gurka MJ (2017). "Independent associations between metabolic syndrome severity and future coronary heart disease by sex and race." *Journal of the American College of Cardiology*, **69**(9), 1204–1205. doi:10.1016/j.jacc.2016.10.088. (clinical application) Gurka MJ, Filipp SL, Pearson TA, DeBoer MD (2018). "Assessing Baseline and Temporal Changes in Cardiometabolic Risk Using Metabolic Syndrome Severity and Common Risk Scores." *Journal of the American Heart Association*, **7**(16), e009754. doi:10.1161/JAHA.118.009754. (clinical application)

Examples

```
df <- data.frame(
  waist = 95, bp_sys = 120, bp_dia = 80, TG = 1.5, HDL_c = 1.2,
  glucose = 5.5, sex = 1, race = "NHW", age = 45
)
metss(df)
```

---

nfl_marker *Neurofilament Light Chain (NfL) Biomarker*

Description

Incorporates a neurofilament light chain (NfL) measurement into the analysis pipeline. Placeholder for future NfL-based computations; returns provided values with input checks.

Usage

```
nfl_marker(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error", "ignore", "warn"),
  verbose = TRUE
)
```

Arguments

data	A data.frame or tibble with an NfL concentration column.
col_map	Named list with nfl indicating the NfL column name.
na_action	One of c("keep","omit","error","ignore","warn").
verbose	Logical; if TRUE (default), emits progress messages.

Details

NfL is released during neuroaxonal injury; elevated levels in CSF or blood indicate neuroaxonal damage and typically increase with age and in neurological diseases. Interpretation requires context-specific and age-adjusted references. This function simply returns the input NfL values (assumed in a single matrix/fluid, e.g., plasma pg/mL) without classification.

Value

A tibble with one column: nfl_value (numeric; same units as input).

References

Simrén J, Ashton NJ, Blennow K, Zetterberg H, et al. (2022). "Reference values for plasma neurofilament light in healthy individuals." *Brain Communications*, **4**(4), fcac174. doi:10.1093/braincomms/fcac174. Disanto G, Barro C, Benkert P, et al. (2017). "Serum neurofilament light: a biomarker of neuronal damage in multiple sclerosis." *Annals of Neurology*, **81**(6), 857–870. doi:10.1002/ana.24954.

Examples

```
df <- data.frame(NfL = c(8.5, 14.2, 22.1))
nfl_marker(df)
```

`normalize_vec` *Normalize a numeric vector*

Description

Utility used across HealthMarkers to normalize numeric vectors with several common schemes while handling edge cases (constant vectors, all-NA, non-finite values) robustly. NA positions are preserved.

Usage

```
normalize_vec(
  x,
  method = c("none", "z", "inverse", "range", "robust"),
  na_rm = TRUE,
  feature_range = c(0, 1),
  invnorm_denominator = c("n", "n+1", "blom"),
  ties = c("average", "first", "last", "random", "max", "min"),
  warn_constant = TRUE
)
```

Arguments

`x`	A numeric (or numeric-coercible) vector.
`method`	One of c("none","z","inverse","range","robust"). Default "none".
`na_rm`	Logical; remove NAs when estimating statistics (mean, sd, etc.). Default TRUE.
`feature_range`	Numeric length-2 vector giving the target range for method = "range". Default c(0, 1).
`invnorm_denominator`	One of c("n","n+1","blom") controlling the denominator of the inverse-normal transform: • "n": p = (r - 0.5) / n (Rankit; default) • "n+1": p = (r - 0.5) / (n + 1) • "blom": p = (r - 3/8) / (n + 1/4) (Blom, 1958)
`ties`	Ties method passed to base::rank for method = "inverse". One of c("average","first","last","random","max Default "average".
`warn_constant`	Logical; if TRUE, warn when input is constant and a zero vector (or lower bound for range) is returned. Default TRUE.

Details

Methods:

- "none": return input as-is (no coercion; fully backward compatible).
- "z": z-score (mean 0, sd 1). Constant vectors return zeros (non-NA entries).

- "range": min-max to a target interval (default $[0, 1]$). Constant vectors return the lower bound (mapped from zeros).
- "robust": median/MAD scaling. Constant vectors (MAD=0) return zeros.
- "inverse": rank-based inverse normal transform (normal scores).

Value

A numeric vector of the same length as x.

References

Beasley TM, Erickson S, Allison DB (2009). "Rank-based inverse normal transformations are increasingly used, but are they merited?" *Behavior Genetics*, **39**(2), 214–227. Leys C, Ley C, Klein O, Bernard P, Licata L (2013). "Detecting outliers: Do not use standard deviation around the mean, use median absolute deviation around the median." *Journal of Experimental Social Psychology*, **49**(4), 764–766. doi:10.1016/j.jesp.2013.03.013. Bland JM, Altman DG (1996). "Statistics notes: measurement error." *BMJ*, **313**(7047), 41–42.

Examples

```
x <- c(1, 2, 3, NA, 5)
normalize_vec(x, "none")
normalize_vec(x, "z")
normalize_vec(x, "range", feature_range = c(-1, 1))
normalize_vec(x, "robust")
normalize_vec(x, "inverse")               # Rankit (default)
normalize_vec(x, "inverse", invnorm_denominator = "blom")
```

---

nutrient_markers *Compute a Suite of Nutrient-Based Health Markers*

Description

Given a data frame or tibble of routine biochemical labs, `nutrient_markers()` returns a set of widely used ratios, products, and simple percentages that summarize iron metabolism, protein status, omega-3 balance, renal excretion, mineral homeostasis, and aromatic amino-acid patterns.

Usage

```
nutrient_markers(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error"),
  na_warn_prop = 0.2,
  verbose = TRUE
)
```

Arguments

data	A data frame or tibble containing subject-level data.
col_map	Optional named list mapping variable keys (see Details) to column names in data. You only need to supply the keys you have; any markers with missing inputs return NA. If NULL, defaults to identity mapping for all known keys.
na_action	One of c("keep","omit","error") controlling missing-data policy across the columns referenced by col_map. • "keep" (default): keep NA; outputs become NA where inputs are NA. • "omit": drop rows with NA in any used input column. • "error": abort if any used input contains NA.
na_warn_prop	Numeric in $[0, 1]$; per-variable threshold for high-missingness diagnostics on used input columns. Default 0.2.
verbose	Logical; if TRUE, prints stepwise messages and a final summary via hm_inform. Default FALSE.

Details

Recognized markers (returned as columns):

- FerritinTS: Ferritin / Transferrin saturation
- AGR: Albumin / Globulin, where Globulin = Total protein - Albumin
- Omega3Index: EPA + DHA (percentage points)
- Mg_Cr_Ratio: Magnesium / Creatinine
- GlycatedAlbuminPct: (Glycated albumin / Albumin) x 100
- UA_Cr_Ratio: Uric acid / Creatinine
- BUN_Cr_Ratio: BUN / Creatinine
- Ca_x_Phosphate: Calcium x Phosphate
- AnionGap: (Na + K) - (Cl + HCO3)
- Tyr_Phe_Ratio: Tyrosine / Phenylalanine

Recognized col_map keys and expected units (no automatic conversion):

- ferritin: Serum ferritin (ng/mL)
- transferrin_sat: Transferrin saturation (%)
- albumin: Serum albumin (g/L)
- total_protein: Total serum protein (g/L)
- EPA: Red-cell EPA as % of total fatty acids
- DHA: Red-cell DHA as % of total fatty acids
- Mg: Serum magnesium (mmol/L)
- creatinine: Serum creatinine (umol/L)
- glycated_albumin: Glycated albumin (g/L)
- uric_acid: Serum uric acid (umol/L)

- BUN: Blood urea nitrogen (mg/dL)
- phosphate: Serum phosphate (mmol/L)
- calcium: Serum calcium (mmol/L)
- Na: Serum sodium (mmol/L)
- K: Serum potassium (mmol/L)
- Cl: Serum chloride (mmol/L)
- HCO3: Serum bicarbonate (mmol/L)
- Tyr: Serum tyrosine (umol/L)
- Phe: Serum phenylalanine (umol/L)

Unit-mixing notes:

- `BUN_Cr_Ratio` divides BUN (mg/dL) by creatinine (umol/L). This is NOT numerically equivalent to the standard clinical BUN:Creatinine ratio (reference range ~10-20), which requires both in mg/dL. The result here is approximately 88.4x smaller than the standard ratio. Provide creatinine in mg/dL and adjust `col_map` if you require the standard clinical ratio.
- `Mg_Cr_Ratio` divides Mg (mmol/L) by creatinine (umol/L). The result is 1/1000 of the standard Mg/Cr ratio in mmol/mmol. Typically applied to urine; serum Mg/Cr is not a standard clinical metric.

Default `extreme_rules` (inputs) are broad and intended for unit/entry checks: ferritin (0, 2000), transferrin_sat (0, 100), albumin (10, 60), total_protein (40, 100), EPA (0, 20), DHA (0, 20), Mg (0.2, 3), creatinine (20, 2000), glycated_albumin (0, 60), uric_acid (50, 1000), BUN (1, 150), phosphate (0.1, 5), calcium (0.5, 4), Na (100, 200), K (2, 8), Cl (70, 130), HCO3 (5, 45), Tyr (10, 300), Phe (20, 300).

Value

A tibble with one row per input row and these columns: FerritinTS, AGR, Omega3Index, Mg_Cr_Ratio, GlycatedAlbuminPct, UA_Cr_Ratio, BUN_Cr_Ratio, Ca_x_Phosphate, AnionGap, Tyr_Phe_Ratio.

References

Harris WS, von Schacky C (2004). "The Omega-3 Index: a new risk factor for death from coronary heart disease?" *Preventive Medicine*, **39**(1), 212–220. doi:10.1016/j.ypmed.2004.02.030. Koga M, Kasayama S (2010). "Clinical impact of glycated albumin as another glycemic control marker." *Endocrine Journal*, **57**(9), 751–762. doi:10.1507/endocrj.k10e138. Block GA, Hulbert-Shearon TE, Levin NW, Port FK (1998). "Association of serum phosphorus and calcium-phosphate product with mortality risk in chronic hemodialysis patients: a national study." *American Journal of Kidney Diseases*, **31**(2), 607–617. doi:10.1053/ajkd.1998.v31.pm9531176. Waikar SS, Bonventre JV (2009). "Creatinine kinetics and the definition of acute kidney injury." *Journal of the American Society of Nephrology*, **20**(3), 672–679. doi:10.1681/ASN.2008070669. (creatinine kinetics context)

Examples

```
# Quick smoke-test
df <- data.frame(ferritin = 50, albumin = 45, uric_acid = 300, Na = 140)
nutrient_markers(df, verbose = FALSE)


df <- tibble::tibble(
  ferritin         = c(50, 100),
  transferrin_sat  = c(30, 50),
  albumin          = c(45, 40),
  total_protein    = c(70, 75),
  EPA              = c(2.0, 2.5),
  DHA              = c(4.0, 4.5),
  Mg               = c(0.85, 0.90),
  creatinine       = c(80, 90),
  glycated_albumin = c(12, 14),
  uric_acid        = c(300, 400),
  BUN              = c(14, 16),
  phosphate        = c(1.0, 1.2),
  calcium          = c(2.3, 2.4),
  Na               = c(140, 138),
  K                = c(4.2, 4.0),
  Cl               = c(100, 102),
  HCO3             = c(24, 26),
  Tyr              = c(60, 70),
  Phe              = c(50, 55)
)
nutrient_markers(df, verbose = TRUE)
```

---

obesity_indices *Compute anthropometric obesity & adiposity indices*

Description

Calculates a comprehensive set of body shape and adiposity indices:

- BMI and WHO BMI categories
- Waist-to-hip ratio (WHR) and optional WHR adjusted for BMI (WHRadjBMI)
- Waist-to-height ratio (WHtR)
- Abdominal Volume Index (AVI)
- Body Adiposity Index (BAI)
- A Body Shape Index (ABSI)
- Body Roundness Index (BRI)
- Conicity Index (CI)
- (Optional) Relative Fat Mass (RFM)

Usage

```
obesity_indices(
  data,
  weight,
  height,
  waist,
  hip,
  sex = NULL,
  weight_unit = c("kg", "lb"),
  height_unit = c("cm", "m"),
  adjust_WHR = FALSE,
  include_RFM = FALSE,
  na_action = c("keep", "omit", "error"),
  na_warn_prop = 0.2,
  verbose = TRUE
)
```

Arguments

data	A data.frame or tibble containing the input columns.
weight	Unquoted column name for weight.
height	Unquoted column name for height.
waist	Unquoted column name for waist circumference.
hip	Unquoted column name for hip circumference.
sex	(Optional) Unquoted column name for sex, coded 0=male, 1=female; required if include_RFM=TRUE.
weight_unit	One of c("kg","lb"); if "lb", converts weight to kg by *0.45359237.
height_unit	One of c("cm","m"); if "cm", converts height to metres by /100.
adjust_WHR	Logical; if TRUE, adds a column WHRadjBMI as residuals from WHR ~ BMI.
include_RFM	Logical; if TRUE, computes Relative Fat Mass (requires sex column).
na_action	One of c("keep","omit","error") for handling NA in required inputs. Default "keep".
na_warn_prop	Proportion $[0, 1]$ to trigger high-missingness warnings for required inputs. Default 0.2.
verbose	Logical; if TRUE, prints column mapping and computing messages.

Details

Units assumed (no automatic conversion beyond the specified weight/height options):

- weight: kg (or lb if weight_unit = "lb")
- height: m (or cm if height_unit = "cm")
- waist, hip: cm

Note: WHtR, ABSI, BRI, CI, and RFM all require waist in the same unit as height (metres). The function converts waist internally (waist_cm / 100) for these five indices; users should always supply waist in cm.

- sex: 0 = male, 1 = female (only required if include_RFM = TRUE)

Value

A tibble with only the computed indices (slim output):

- weight_kg, height_m (unit-normalised intermediates),
- BMI, BMI_cat,
- WHR, WHRadjBMI (if `adjust_WHR = TRUE`),
- waist_to_height_ratio, waist_to_BMI_ratio, weight_to_height_ratio,
- AVI, BAI, ABSI, BRI, CI,
- RFM (if `include_RFM = TRUE`).

References

Quetelet A (1842). *A Treatise on Man, and the Development of his Faculties*. William and Robert Chambers, Edinburgh. Historical monograph; no DOI assigned, https://archive.org/search?query=A%20Treatise%20on%20Man%20Quetelet. WHO Expert Committee (1995). "Physical Status: The Use and Interpretation of Anthropometry. WHO Technical Report Series 854." World Health Organization, Geneva. No DOI for this WHO technical report; see URL, https://www.who.int/publications/i/item/9241208546. Guerrero-Romero F, Rodríguez-Morán M (2003). "Abdominal volume index. An anthropometric index of central obesity." *Archives of Medical Research*, **34**(6), 428–432. doi:10.1016/S01884409(03)000730. Bergman RN, Stefanovski D, Buchanan TA, others (2011). "A better index of body adiposity." *Obesity (Silver Spring)*, **19**(5), 1083–1089. doi:10.1038/oby.2011.38. Krakauer NY, Krakauer JC (2012). "A new body shape index predicts mortality hazard independently of BMI." *PLoS One*, **7**(7), e39504. doi:10.1371/journal.pone.0039504. Thomas DM, Bredlau C, Bosy-Westphal A, others (2013). "Relationships between body roundness with body fat and visceral adipose tissue emerging from a new geometrical model." *Obesity (Silver Spring)*, **21**(11), 2264–2271. doi:10.1002/oby.20408. Valdez R (1991). "A simple model-based index of abdominal adiposity." *Journal of Clinical Epidemiology*, **44**(9), 955–956. doi:10.1016/08954356(91)90059I. Woolcott OO, Bergman RN (2018). "Relative fat mass (RFM) as a new estimator of whole-body fat percentage." *Scientific Reports*, **8**, 10980. doi:10.1038/s41598018293621. Calle EE, Thun MJ, Petrelli JM, Rodriguez C, Heath Jr. CW (1999). "Body-mass index and mortality in a prospective cohort of U.S. adults." *New England Journal of Medicine*, **341**(15), 1097–1105. doi:10.1056/NEJM199910073411501. Freedman DS, Thornton JC, Pi-Sunyer FX, others (2012). "The body adiposity index is not a more accurate measure of adiposity than BMI, waist circumference, or hip circumference." *Obesity (Silver Spring)*, **20**(12), 2438–2444. doi:10.1038/oby.2012.81. He S, Chen X (2013). "Could the new body shape index predict the new onset of diabetes mellitus in the Chinese population?" *PLoS One*, **8**(1), e50573. doi:10.1371/journal.pone.0050573. Maessen MF, Eijsvogels TM, Verheggen RJ, others (2014). "Entering a new era of body indices: the feasibility of ABSI and BRI to identify cardiovascular health status." *PLoS One*, **9**(9), e107212. doi:10.1371/journal.pone.0107212.

Examples

```
library(tibble)
df <- tibble(
  wt     = c(70, 80),  # kg
  ht     = c(175, 165),# cm
  waist  = c(80, 90),  # cm
  hip    = c(100, 95), # cm
  sex    = c(0, 1)
)
obesity_indices(
  df,
  weight       = wt,
  height       = ht,
  waist        = waist,
  hip          = hip,
  sex          = sex,
  weight_unit  = "kg",
  height_unit  = "cm",
  adjust_WHR   = TRUE,
  include_RFM  = TRUE,
  verbose      = TRUE
)
```

`ogtt_is` *Calculate OGTT-based insulin sensitivity indices*

Description

Given glucose & insulin at 0, 30, 120 min (plus weight, BMI, age, sex), computes:

- Isi_120
- Cederholm_index
- Gutt_index
- Avignon_Si0
- Avignon_Si120
- Avignon_Sim
- Modified_stumvoll
- Stumvoll_Demographics
- Matsuda_AUC
- Matsuda_ISI
- BigttSi
- Ifc_inv
- HIRI_inv
- Belfiore_isi_gly

Usage

```
ogtt_is(
  data,
  col_map = NULL,
  normalize = "none",
  verbose = TRUE,
  na_action = c("keep", "omit", "error"),
  na_warn_prop = 0.2
)
```

Arguments

`data`	A data.frame or tibble containing at least the columns mapped by `col_map`.
`col_map`	Named list mapping: • G0, G30, G120 -> glucose at 0, 30, 120 min (mmol/L) • I0, I30, I120 -> insulin at 0, 30, 120 min (pmol/L) • weight -> body weight (kg) • bmi -> body-mass index (kg/m^2) • age -> age (years) • sex -> sex (1 = male, 2 = female)
`normalize`	One of c("none","z","inverse","range","robust") used by normalize_vec().
`verbose`	Logical; if `TRUE` (default), prints column mapping, the list of indices being computed, and a per-column results summary.
`na_action`	One of c("keep","omit","error") for missing/non-finite required inputs. Default "keep".
`na_warn_prop`	Proportion (0-1) for high-missingness diagnostics (debug). Default 0.2.

Details

Units assumed:

- OGTT glucose in mmol/L (internally converted to mg/dL via *18 for select indices)
- OGTT insulin in pmol/L (internally converted to muU/mL via /6 for select indices)
- weight in kg; BMI in kg/m^2; age in years; sex coded 1 = male, 2 = female

Notes

- Conversions mirror existing implementation to preserve outputs. Some formulas intentionally use unconverted inputs (as in prior code).
- Modified_stumvoll and Stumvoll_Demographics use raw pmol/L and mmol/L as published in Stumvoll et al. (2000). BigttSi likewise uses raw units.
- Matsuda_AUC is a non-standard AUC-based variant; the original Matsuda index (Matsuda_ISI) uses time-point means, not AUCs.
- Cederholm_index uses log(I0 + I120) as implemented (sum, not mean); Gutt_index uses log((I0 + I120)/2) (mean). This mirrors published implementations; the difference is a constant log(2) offset.

- Ifc_inv and HIRI_inv are derived composite proxies not attributed to a single formula publication; treat as research tools.
- Logs are safe: log(x) becomes NA when x <= 0 or non-finite.

Value

A tibble with the OGTT-based index columns listed above. If an ID column is detected in `data` (e.g. `id`, `IID`, `participant_id`), it is prepended as the first output column.

References

Matsuda M, DeFronzo RA (1999). "Insulin Sensitivity Indices Obtained from Oral Glucose Tolerance Testing: Comparison with the Minimal Model Assessment." *Diabetes Care*, **22**(9), 1462–1470. doi:10.2337/diacare.22.9.1462. Gutt M, Davis CL, Spitzer SB, et al. (2000). "Validation of the Insulin Sensitivity Index ($ISI_{0,120}$) Derived from Oral Glucose Tolerance Testing." *Diabetes Research and Clinical Practice*, **47**(3), 177–184. doi:10.1016/S01688227(99)001163. Stumvoll M, Mitrakou A, Pimenta W, et al. (2000). "Use of the Oral Glucose Tolerance Test to Assess Insulin Release and Sensitivity." *Diabetes Care*, **23**(3), 295–301. doi:10.2337/diacare.23.3.295. Hansen T, Drivsholm T, Urhammer SA, Palacios RR, et al. (2007). "The BIGTT Test." *Diabetes Care*, **30**(2), 257–262. doi:10.2337/dc061240. Avignon A, Charles M, Rabasa-Lhoret R, et al. (1999). "Assessment of Insulin Sensitivity from Oral Glucose Tolerance Test in Normal Subjects and in Insulin-Resistant Patients." *International Journal of Obesity*, **23**(5), 512–517. doi:10.1038/sj.ijo.0800864. Belfiore F, Iannello S, Volpicelli G (1998). "Insulin Sensitivity Indices Calculated from Basal and OGTT-Related Insulin and Glucose Levels." *Molecular Genetics and Metabolism*, **63**(2), 134–141. doi:10.1006/mgme.1997.2658. Matthews DR, Hosker JP, Rudenski AS, Naylor BA, Treacher DF, Turner RC (1985). "Homeostasis Model Assessment: Insulin Resistance and Beta-Cell Function from Fasting Plasma Glucose and Insulin Concentrations in Man." *Diabetologia*, **28**(7), 412–419. doi:10.1007/BF00280883. Suleman S, Madsen AL, Ängquist LH, Schubert M, Linneberg A, Loos RJF, Hansen T, Grarup N (2024). "Genetic Underpinnings of Fasting and Oral Glucose-stimulated Based Insulin Sensitivity Indices." *The Journal of Clinical Endocrinology & Metabolism*, **109**(11), 2754–2763. doi:10.1210/clinem/dgae275. (genetic epidemiology study reviewing IS indices)

See Also

`fasting_is()`, `normalize_vec()`

Examples

```
df <- tibble::tibble(
  G0 = 5.5, I0 = 60,
  G30 = 7.8, I30 = 90,
  G120 = 6.2, I120 = 50,
  weight = 70, bmi = 24, age = 30, sex = 1
)
ogtt_is(
  df,
  col_map = list(
    G0 = "G0", I0 = "I0",
    G30 = "G30", I30 = "I30",
    G120 = "G120", I120 = "I120",
```

```
    weight = "weight", bmi = "bmi",
    age = "age", sex = "sex"
  ),
  normalize = "none",
  verbose = TRUE
)
```

`oxidative_markers` *Oxidative stress markers*

Description

Computes GSH_GSSG_Ratio = reduced glutathione (GSH) / oxidized glutathione (GSSG).

Usage

```
oxidative_markers(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error"),
  verbose = TRUE
)
```

Arguments

`data`	Data frame with columns for GSH and GSSG (per col_map).
`col_map`	Named list with required keys `GSH` and `GSSG`. Defaults assume column names match keys. Both columns must be in the same units (e.g., µmol/L).
`na_action`	One of c("keep","omit","error").
`verbose`	Logical; if `TRUE` (default), prints column mapping and a per-column results summary.

Value

A tibble with column GSH_GSSG_Ratio. If an ID column is detected, it is prepended.

Note

`GSH_GSSG_Ratio` is dimensionless only when `GSH` and `GSSG` are supplied in the same units (typically µmol/L). The formula `GSH / GSSG` is a standard biochemical redox ratio; no unit conversion is applied.

References

Forman HJ, Zhang H, Rinna A (2009). "Glutathione: Overview of its protective roles, measurement, and biosynthesis." *Molecular Aspects of Medicine*, **30**(1–2), 1–12. doi:10.1016/j.mam.2008.08.006. (background review; GSH/GSSG is a standard biochemical redox ratio, not a formula from this paper)

Examples

```
df <- data.frame(GSH = c(5, 3), GSSG = c(1, 0.5))
oxidative_markers(df, col_map = list(GSH="GSH", GSSG="GSSG"))
```

`phq9_score` *PHQ-9 / PHQ-8 scoring*

Description

PHQ-9 / PHQ-8 scoring

Usage

```
phq9_score(
  data,
  col_map = list(),
  na_action = c("keep", "omit", "error"),
  missing_prop_max = 0.2,
  impute = c("none", "mean"),
  variant = c("PHQ9", "PHQ8"),
  prefix = "PHQ9",
  verbose = TRUE
)
```

Arguments

`data`	Data frame containing questionnaire item columns.
`col_map`	Named list mapping canonical item IDs to column names; defaults assume items are already named.
`na_action`	How to handle rows with missing items: `keep`, `omit`, or `error`.
`missing_prop_max`	Maximum allowed proportion of missing items per row before the score is set to `NA`.
`impute`	Imputation strategy for missing items when under the threshold: `none` or `mean` (row-wise mean).
`variant`	Choose PHQ9 (9 items) or PHQ8 (drops suicidal ideation item).
`prefix`	Prefix for output column names.
`verbose`	Logical; if `TRUE`, emits informational messages about column resolution and scoring progress via `hm_inform()`.

Value

A tibble of score columns only: `PHQ9_total` and `PHQ9_severity` (factor). Input columns are not included.

References

Kroenke K, Spitzer RL, Williams JBW (2001). "The PHQ-9: Validity of a Brief Depression Severity Measure." *Journal of General Internal Medicine*, **16**(9), 606–613. doi:10.1046/j.1525-1497.2001.016009606.x.

Examples

```
df <- data.frame(phq9_01 = 0, phq9_02 = 1, phq9_03 = 2, phq9_04 = 1,
                 phq9_05 = 0, phq9_06 = 1, phq9_07 = 2, phq9_08 = 1,
                 phq9_09 = 0)
phq9_score(df)
```

---

plot_frailty_age *Plot FI vs age (convenience wrapper)*

Description

Calls frailty_index() with visible = TRUE; see frailty_index() for arguments, validation, and references.

Usage

```
plot_frailty_age(
  data,
  cols = NULL,
  invert = NULL,
  rescale = TRUE,
  age = NULL,
  rescale.custom = NULL,
  rescale.avoid = NULL,
  bins = 7,
  na_action = c("ignore", "warn", "error", "keep", "omit"),
  na_warn_prop = 0.2,
  return = c("list", "data"),
  verbose = TRUE
)
```

Arguments

data	A data.frame or tibble of health deficits (ideally binary/logical or scaled to $[0, 1]$). Non-binary numeric columns can be rescaled by di::di when rescale = TRUE.
cols	Character vector of deficit column names to use. If NULL (default), all numeric columns are used except age (if supplied).
invert	Character vector of column names whose values should be inverted by di::di (e.g., where higher values indicate better health).

rescale	Logical; if TRUE, non-binary columns will be rescaled to $[0, 1]$ by di::di. Default TRUE.
age	Optional name of the column holding age (used by di for plotting and optional age-binned outputs; excluded from auto-selected cols).
rescale.custom	Advanced argument passed through to di::di. See di::di documentation for syntax.
rescale.avoid	Advanced argument passed through to di::di; see di::di documentation for syntax.
bins	Integer; number of age bins for FI-by-age plots. Default 7.
na_action	One of c("keep","omit","error","warn","ignore"). Controls handling of missing values in selected deficit columns. "keep" (and its backward-compatible alias "ignore") passes NAs through to di::di. "warn" emits a warning and then keeps NAs (alias for "keep" with a missingness warning). "omit" drops rows with any NA in selected deficits before computing. "error" stops if any NA is detected. Default "ignore" (retained for backward compatibility; equivalent to "keep").
na_warn_prop	Proportion in $[0, 1]$ above which a high-missingness warning is emitted (per column) when na_action = "warn". Default 0.2.
return	One of c("list","data"). "list" (default) returns the original di::di result (backward compatible). "data" returns a tibble with one row per individual, columns: di (the frailty index) plus the selected deficit columns (post-capping if applied). Age is included if present.
verbose	Logical; if TRUE, prints progress and a completion summary.

Value

The object returned by frailty_index() (di::di object if return="list").

Examples

```
if (requireNamespace("di", quietly = TRUE)) {
  df <- data.frame(age = c(70, 75, 80), d1 = c(0, 1, 1),
    d2 = c(0.2, 0.8, 1.0), d3 = c(TRUE, FALSE, TRUE))
  plot_frailty_age(df, cols = c("d1", "d2", "d3"), age = "age")
}
```

---

psych_dx_flags *Psychiatric diagnosis flags aggregator*

Description

Psychiatric diagnosis flags aggregator

Usage

```
psych_dx_flags(
  data,
  col_map = list(),
  prefix = "dx",
  na_action = c("keep", "omit", "error")
)
```

Arguments

data	Data frame containing questionnaire item columns.
col_map	Named list dx mapping condition ids (e.g., mdd, bipolar) to columns of boolean/numeric flags.
prefix	Prefix for output flag columns.
na_action	How to handle rows with missing items: keep, omit, or error.

Value

A tibble of flag columns only: dx_any_psych, dx_internalizing, dx_externalizing, dx_psychotic, dx_count. Input columns are not included.

Examples

```
df <- data.frame(dx_mdd = c(1, 0), dx_bipolar = c(0, 1))
psych_dx_flags(df, col_map = list(dx = list(mdd = "dx_mdd", bipolar = "dx_bipolar")))
```

---

psych_markers *Psychometric markers dispatcher*

Description

Psychometric markers dispatcher

Usage

```
psych_markers(
  data,
  col_map = list(),
  which = c("phq9", "gad7", "k6", "k10", "ghq12_likert", "ghq12_binary", "who5", "isi",
    "mdq", "asrs", "bis", "spq", "cognitive", "dx_flags", "med_flags"),
  na_action = c("keep", "omit", "error"),
  missing_prop_max = 0.2,
  impute = c("none", "mean"),
  bis_key = NULL,
  spq_key = NULL,
  cognitive_method = c("z_mean", "pca1"),
  verbose = TRUE
)
```

Arguments

data	Data frame containing questionnaire item columns.
col_map	Nested list of mappings per instrument (e.g., col_map$phq9, col_map$bis, col_map$dx_flags, ...).
which	Vector of modules to compute (e.g., "phq9", "gad7", "bis").
na_action	How to handle rows with missing items: keep, omit, or error.
missing_prop_max	Maximum allowed proportion of missing items per row before the score is set to NA.
impute	Imputation strategy for missing items when under the threshold: none or mean (row-wise mean).
bis_key	SPQ/BIS key list passed to bis_score when requested.
spq_key	SPQ key list passed to spq_score when requested.
cognitive_method	Method passed to cognitive_score ("z_mean" or "pca1").
verbose	Logical; if TRUE, emits informational messages about column resolution and scoring progress via hm_inform().

Value

A tibble of computed score columns from all requested modules, bound together. No input columns are included in the output.

Examples

```
df <- data.frame(
  phq9_01 = 0, phq9_02 = 1, phq9_03 = 2, phq9_04 = 1, phq9_05 = 0,
  phq9_06 = 1, phq9_07 = 2, phq9_08 = 1, phq9_09 = 0,
  gad7_01 = 0, gad7_02 = 1, gad7_03 = 2, gad7_04 = 1, gad7_05 = 0,
  gad7_06 = 1, gad7_07 = 2
)
psych_markers(df, which = c("phq9", "gad7"))
```

---

psych_med_flags *Psychiatric medication flags aggregator*

Description

Psychiatric medication flags aggregator

Usage

```
psych_med_flags(
  data,
  col_map = list(),
  prefix = "med",
  na_action = c("keep", "omit", "error")
)
```

Arguments

data	Data frame containing questionnaire item columns.
col_map	Named list med mapping medication classes (e.g., ssri, snri) to columns of boolean/numeric flags.
prefix	Prefix for output flag columns.
na_action	How to handle rows with missing items: keep, omit, or error.

Value

A tibble of flag columns only: med_any_psych, med_count. Input columns are not included.

Examples

```
df <- data.frame(med_ssri = c(1, 0), med_antipsychotic = c(0, 1))
cm <- list(med = list(
  ssri = "med_ssri",
  antipsychotic = "med_antipsychotic"
))
psych_med_flags(df, col_map = cm)
```

---

pulmo_markers *Calculate pulmonary function markers (FEV1/FVC, z-scores, percent predicted, LLN, etc.)*

Description

Uses the rspiro reference equations to compute predicted normals, z-scores, percent predicted and lower limits of normal (LLN) for FEV1, FVC, and the FEV1/FVC ratio.

Usage

```
pulmo_markers(
  data,
  col_map = NULL,
  equation = c("GLI", "GLIgl", "NHANES3"),
  na_action = c("keep", "omit", "error"),
  na_warn_prop = 0.2,
  verbose = TRUE
)
```

Arguments

data	A data.frame or tibble with columns: • age (numeric): years • sex (character or numeric): "male"/"female" (case-insensitive) or codes 1/2 or 0/1 • height (numeric): cm or m (auto-detected) • ethnicity (character): e.g. "Caucasian", "African-American", "NE Asian", "SE Asian", "Other/Mixed" • fev1 (numeric): observed FEV1 in L • fvc (numeric): observed FVC in L
col_map	Optional named list mapping canonical keys (age, sex, height, ethnicity, fev1, fvc) to actual column names in data. If NULL, column names are inferred automatically.
equation	One of c("GLI","GLIgl","NHANES3") (see rspiro for details). GLIgl ignores ethnicity.
na_action	One of c("keep","omit","error") for handling missing values in required inputs. Default "keep".
na_warn_prop	Proportion $[0, 1]$ to trigger high-missingness warnings on required inputs. Default 0.2.
verbose	Logical; if TRUE prints progress messages and a completion summary.

Details

Inputs are validated, missingness is handled via na_action, and heights are auto-detected as cm when any height > 3; otherwise interpreted as metres (no automatic unit conversion beyond that heuristic, preserving prior behavior).

Value

A tibble with columns:

- fev1_pred, fev1_z, fev1_pctpred, fev1_LLN
- fvc_pred, fvc_z, fvc_pctpred, fvc_LLN
- fev1_fvc_ratio, fev1_fvc_pred, fev1_fvc_z, fev1_fvc_pctpred, fev1_fvc_LLN (NA if the equation lacks native FEV1/FVC support in rspiro)

Note

fev1_fvc_z, fev1_fvc_pred, and fev1_fvc_LLN are computed via rspiro's native FEV1/FVC parameter (equivalent to param = "FEV1FVC"). If that parameter is not supported by the installed rspiro version or equation, these columns fall back gracefully: fev1_fvc_z and fev1_fvc_LLN become NA; fev1_fvc_pred falls back to fev1_pred / fvc_pred.

References

Quanjer PH, Stanojevic S, Cole TJ, Baur X, Hall GL, Culver BH, et al. (2012). "Multi-ethnic reference values for spirometry for the 3–95-yr age range: the global lung function 2012 equations." *European Respiratory Journal*, **40**, 1324–1343. doi:10.1183/09031936.00080312. Hankinson JL, Odencrantz JR, Fedan KB (1999). "Spirometric reference values from a sample of the general U.S. population." *American Journal of Respiratory and Critical Care Medicine*, **159**, 179–187. doi:10.1164/ajrccm.159.1.9712108. Stanojevic S, Kaminsky DA, Miller MR, Thompson BR, Aliverti A, Barjaktarevic I, Cooper BG, Culver BH, Derom E, Hall GL, Heggie A, Iyer VN, Jackson AS, Jensen RL, Langer D, Latourelle JC, Laucho-Contreras ME, MacIntyre N, McCormack MC, Rosenfeld M, Swenson ER, Thompson C, Topalovic M, Wilsher M, Wijnant SRA, Gore R, Ramsey SM, Bhatt SP (2022). "ERS/ATS technical standard on interpretive strategies for routine lung function tests." *European Respiratory Journal*, **60**(1), 2101499. doi:10.1183/13993003.014992021. (race-neutral GLI global equations and interpretation framework; used by rspiro's `GLIgl` equation)

See Also

rspiro

Examples

```
if (requireNamespace("rspiro", quietly = TRUE)) {
  df <- data.frame(
    age = c(40, 55), sex = c("male", "female"),
    height = c(175, 162), ethnicity = c("Caucasian", "Caucasian"),
    fev1 = c(3.5, 2.4), fvc = c(4.4, 3.1)
  )
  pulmo_markers(df)
}
```

`renal_markers` *Calculate a Suite of Renal Function, Injury, and Excretion Markers*

Description

Given routine blood and urine assays, `renal_markers()` computes:

- eGFR_cr: CKD-EPI creatinine equation (2009 variant; race factor retained to preserve prior behavior)
- eGFR_cys: CKD-EPI cystatin C equation (if `cystatin_C` provided)
- eGFR_combined: CKD-EPI combined creatinine+cystatin C (if both provided)
- BUN_Cr_ratio: Blood urea nitrogen / serum creatinine
- FE_Urea: Fractional excretion of urea (%)
- NGAL, KIM1, NAG, Beta2Micro, IL18, L_FABP: pass-through urinary injury markers (if mapped)

Usage

```
renal_markers(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error"),
  na_warn_prop = 0.2,
  verbose = TRUE
)
```

Arguments

data	A data.frame or tibble with renal lab data.
col_map	Named list mapping: • creatinine -> serum creatinine (mg/dL) • age -> age (years) • sex -> sex indicator (1 = male, 0 = female). Also accepts "male"/"female". • race -> race ("white", "black", or "other"). Also accepts common aliases. • BUN -> blood urea nitrogen (mg/dL) • optional cystatin_C -> serum cystatin C (mg/L) • optional urea_serum -> serum urea (mg/dL) • optional creatinine_urine -> urine creatinine (mg/dL) • optional urea_urine -> urine urea (mg/dL) • optional NGAL, KIM1, NAG, beta2_micro, IL18, L_FABP -> urine injury markers
na_action	One of c("keep","omit","error") for handling missing values in required inputs. Default "keep".
na_warn_prop	Proportion (0-1) threshold for high-missingness diagnostics. Default 0.2.
verbose	Logical; if TRUE (default), prints step-by-step progress including column mapping, optional input availability, physiological range information (informational only, values are not altered), the list of markers being computed, and a per-column results summary.

Details

Robust validation is applied, including NA handling (na_action), high-missingness diagnostics, safe divisions with a consolidated zero-denominator warning, and an optional input extremes scan/cap. New arguments are appended for backward compatibility.

Expected units (no automatic conversion performed):

- creatinine (serum): mg/dL
- cystatin C (serum): mg/L
- BUN (serum): mg/dL
- urea_serum, urea_urine: mg/dL
- creatinine_urine: mg/dL

Value

A tibble with computed renal markers: `eGFR_cr`, `eGFR_cys`, `eGFR_combined`, `BUN_Cr_ratio`, `FE_Urea`, `NGAL`, `KIM1`, `NAG`, `Beta2Micro`, `IL18`, `L_FABP`. If an ID column is detected in `data` (e.g. `id`, `IID`, `participant_id`), it is prepended as the first output column.

Note

`eGFR_cr` uses the 2009 CKD-EPI creatinine equation (Levey et al. 2009) with the Black-race multiplier ($\times 1.159$) retained. The 2021 race-free CKD-EPI equations (Inker et al., NEJM 2021) are not yet implemented; the `race` input is accepted for forward compatibility and used only for the 2009 race factor. `eGFR_cys` and `eGFR_combined` use Inker et al. (2012); note that `eGFR_combined` applies its own sex ($\times 1.008$ female) and race ($\times 1.145$ Black) multipliers, which differ from those of `eGFR_cr`. `NGAL`, `KIM1`, `NAG`, `Beta2Micro`, `IL18`, and `L_FABP` are **pass-through** columns — values are returned as-is with no formula applied.

References

Levey AS, Stevens LA, Schmid CH, others (2009). "A new equation to estimate glomerular filtration rate." *Annals of Internal Medicine*, **150**(9), 604–612. doi:10.7326/0003481915092009050-00006. Inker LA, Schmid CH, Tighiouart H, others (2012). "Estimating glomerular filtration rate from serum cystatin C." *New England Journal of Medicine*, **367**(1), 20–29. doi:10.1056/NEJMoa1114248. Kaplan AA, Kohn OF (1992). "Fractional Excretion of Urea as a Guide to Renal Dysfunction." *American Journal of Nephrology*, **12**(1–2), 49–54. doi:10.1159/000168417. (FE_Urea formula source; bib content: Kaplan and Kohn 1992) Parikh CR, Coca SG, Thiessen-Philbrook H, others (2011). "Postoperative Biomarkers Predict Acute Kidney Injury and Poor Outcomes after Adult Cardiac Surgery." *Journal of the American Society of Nephrology*, **22**(12), 1737–1747. doi:10.1681/ASN.2010121302. (clinical context; NGAL is a pass-through biomarker) Vaidya VS, Ramirez V, Ichimura T, others (2010). "Kidney injury molecule-1 outperforms traditional biomarkers of kidney injury in preclinical biomarker qualification studies." *Nature Biotechnology*, **28**(5), 478–485. doi:10.1038/nbt.1623. (KIM-1 biomarker qualification; pass-through) Portilla D, Dent C, Sugaya T, others (2008). "Urinary liver-type fatty acid-binding protein as a biomarker of acute kidney injury." *Kidney International*, **73**(4), 465–472. doi:10.1038/sj.ki.5002721. (L-FABP as AKI biomarker; pass-through)

Examples

```
df <- tibble::tibble(Cr = 1.0, Age = 40, Sex = 1, Race = "white", BUN = 14)
cm <- list(creatinine = "Cr", age = "Age", sex = "Sex", race = "Race", BUN = "BUN")
renal_markers(df, cm)
```

`saliva_markers` *Calculate saliva-based stress & glycemic markers*

Description

Computes:

- log_cortisol_wake (log-transformed waking cortisol)
- CAR_AUC (Cortisol Awakening Response, trapezoidal AUC over 0-60 min by default)
- log_amylase (log-transformed salivary alpha-amylase)
- saliva_glucose (raw salivary glucose)

Usage

```
saliva_markers(
  data,
  col_map = NULL,
  verbose = TRUE,
  na_action = c("keep", "omit", "error"),
  na_warn_prop = 0.2,
  times = c(0, 30, 60)
)
```

Arguments

`data`	A data.frame or tibble with salivary markers.
`col_map`	Named list mapping required inputs. Defaults assume same names: • cort1 -> "saliva_cort1" (nmol/L at wake) • cort2 -> "saliva_cort2" (nmol/L ~30 min) • cort3 -> "saliva_cort3" (nmol/L ~60 min) • amylase -> "saliva_amylase" (U/mL) • glucose -> "saliva_glucose" (mg/dL)
`verbose`	Logical; if `TRUE` (default), prints column mapping, input availability, physiological range information (informational only, values not altered), the list of markers being computed with their inputs, and a per-column results summary.
`na_action`	One of `c("keep","omit","error")` for required-input NA handling. Default "keep".
`na_warn_prop`	Proportion $[0, 1]$ to trigger high-missingness diagnostics (debug). Default 0.2.
`times`	Numeric vector of sampling times (minutes) for CAR AUC. Must align with cort1/2/3. Default c(0,30,60).

Details

Inputs are validated, missingness handled via `na_action`, logs made safe (<= 0 -> NA), and optional extremes scan/cap is available.

Value

A tibble with columns:

- log_cortisol_wake
- CAR_AUC
- log_amylase
- saliva_glucose

If an ID column is detected in data (e.g. id, IID, participant_id), it is prepended as the first output column.

Note

log_cortisol_wake and log_amylase use the **natural logarithm** (log()). CAR_AUC is the trapezoidal area under the cortisol-time curve (Pruessner et al. 2003, AUC with respect to ground). saliva_glucose is a **pass-through** column; no formula is applied.

References

Pruessner JC, Kirschbaum C, Meinlschmid G, Hellhammer DH (2003). "Two formulas for computation of the area under the curve represent measures of total hormone concentration versus time-dependent change." *Psychoneuroendocrinology*, **28**(7), 916–931. doi:10.1016/S03064530(02)00108-7. Kirschbaum C, Hellhammer DH (1994). "Salivary cortisol in psychoneuroendocrine research: recent developments and applications." *Psychoneuroendocrinology*, **19**(4), 313–333. doi:10.1016/03064530(94)900132. (salivary cortisol methods; background) Clow A, Thorn L, Evans P, Hucklebridge F (2004). "The awakening cortisol response: methodological issues and significance." *Stress*, **7**(1), 29–37. doi:10.1080/10253890410001667205. (CAR methodological review; background) Nater UM, Rohleder N (2009). "Salivary alpha-amylase as a non-invasive biomarker for the sympathetic nervous system: current state of research." *Psychoneuroendocrinology*, **34**(4), 486–496. doi:10.1016/j.psyneuen.2009.01.014. (salivary alpha-amylase SNS biomarker; background) Scales WE, Freeman EW, McCoy NL, Klerman EB (1987). "Salivary glucose as a measure of blood glucose: correlations and applications." *Diabetes Care*, **10**(4), 414–418. doi:10.2337/diacare.10.4.414. (salivary glucose application; pass-through, no formula)

Examples

```
df <- tibble::tibble(
  saliva_cort1    = 12.5,
  saliva_cort2    = 18.0,
  saliva_cort3    = 16.2,
  saliva_amylase  = 85,
  saliva_glucose  = 4.2
)
saliva_markers(df)  # uses default col_map
```

`sarc_f_score` *SARC-F Sarcopenia Screening Score*

Description

Computes the SARC-F questionnaire score, a quick screening tool for sarcopenia risk.

Usage

```
sarc_f_score(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error", "ignore", "warn"),
  verbose = TRUE
)
```

Arguments

`data`	A data.frame or tibble with SARC-F questionnaire responses.
`col_map`	Named list mapping the five SARC-F components to columns: strength, walking, chair, stairs, falls.
`na_action`	One of c("keep","omit","error","ignore","warn").
`verbose`	Logical; if TRUE (default), emits progress messages.

Details

SARC-F has 5 items: Strength, Assistance in walking, Rise from a chair, Climb stairs, and Falls. Each item is scored 0 (no difficulty) to 2 (high difficulty). Total SARC-F score ranges 0-10. A score >= 4 indicates high risk of sarcopenia and suggests further assessment.

Value

A tibble with:

- sarc_f_score (numeric 0-10; NA if any component is NA)
- sarc_f_high_risk (logical; TRUE if score >= 4, NA if score is NA)

References

Malmstrom TK, Morley JE (2013). "SARC-F: a simple questionnaire to rapidly diagnose sarcopenia." *Journal of the American Medical Directors Association*, **14**(8), 531–532. doi:10.1016/j.jamda.2013.05.018. Malmstrom TK, Miller DK, Simonsick EM, Ferrucci L, Morley JE (2016). "SARC-F: a symptom score to predict persons with sarcopenia at risk for poor functional outcomes." *Journal of Cachexia, Sarcopenia and Muscle*, **7**(1), 28–36. doi:10.1002/jcsm.12048. (SARC-F validation and functional outcome prediction; background)

Examples

```
df <- data.frame(Strength = c(1, 2, 0), Walking = c(0, 1, 2),
                 Chair = c(1, 1, 2), Stairs = c(0, 2, 2), Falls = c(0, 1, 1))
sarc_f_score(df)
```

`spirometry_markers` *Spirometry markers: FEV1/FVC, LLN-based obstruction, GOLD grade, bronchodilator response*

Description

Spirometry markers: FEV1/FVC, LLN-based obstruction, GOLD grade, bronchodilator response

Usage

```
spirometry_markers(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error", "ignore", "warn"),
  verbose = TRUE
)
```

Arguments

`data`	Data frame with spirometry inputs.
`col_map`	Named list: fev1, fvc, fev1_post, fvc_post, age, height, sex, ethnicity
`na_action`	One of c("keep","omit","error","ignore","warn").
`verbose`	Logical; if TRUE (default), emits progress via rlang::inform.

Value

Tibble with ratio_pre, ratio_post, copd_flag_fixed, obstruction_lln, fev1_pp, fvc_pp, fev1_z, fvc_z, ratio_z, gold_grade, bdr_fev1, bdr_fvc.

References

Miller MR, Hankinson J, Brusasco V, et al. (2005). "Standardisation of spirometry." *European Respiratory Journal*, **26**(2), 319–338. doi:10.1183/09031936.05.00034805. (spirometry standardisation methodology; background) Quanjer PH, Stanojevic S, Cole TJ, Baur X, Hall GL, Culver BH, et al. (2012). "Multi-ethnic reference values for spirometry for the 3–95-yr age range: the global lung function 2012 equations." *European Respiratory Journal*, **40**, 1324–1343. doi:10.1183/09031936.00080312. for Chronic Obstructive Lung Disease (GOLD) GI (2025). "Global strategy for the diagnosis, management, and prevention of COPD." Online report; no DOI assigned, https://goldcopd.org/2025-gold-report/.

Examples

```
df <- data.frame(FEV1 = c(3.2, 2.1, 1.5), FVC = c(4.0, 3.0, 2.5))
spirometry_markers(df)
```

`spq_score` *Schizotypal Personality Questionnaire (key-driven)*

Description

Schizotypal Personality Questionnaire (key-driven)

Usage

```
spq_score(
  data,
  col_map = list(),
  key,
  na_action = c("keep", "omit", "error"),
  missing_prop_max = 0.2,
  impute = c("none", "mean"),
  prefix = "SPQ",
  verbose = TRUE
)
```

Arguments

`data`	Data frame containing questionnaire item columns.
`col_map`	Named list mapping canonical item IDs to column names; defaults assume items are already named.
`key`	List with `items`, `min_val`, `max_val`, optional `reverse` and `subscales`.
`na_action`	How to handle rows with missing items: `keep`, `omit`, or `error`.
`missing_prop_max`	Maximum allowed proportion of missing items per row before the score is set to `NA`.
`impute`	Imputation strategy for missing items when under the threshold: `none` or `mean` (row-wise mean).
`prefix`	Prefix for output column names.
`verbose`	Logical; if `TRUE`, emits informational messages about column resolution and scoring progress via `hm_inform()`.

Value

A tibble of score columns only (total and optional subscales). Input columns are not included.

References

Raine A (1991). "The SPQ: A Scale for the Assessment of Schizotypal Personality Based on DSM-III-R Criteria." *Schizophrenia Bulletin*, **17**(4), 555–564. doi:10.1093/schbul/17.4.555.

Examples

```
spq_key <- list(items = sprintf("spq_%02d", 1:5), min_val = 0, max_val = 1)
df <- data.frame(spq_01 = 0, spq_02 = 1, spq_03 = 0, spq_04 = 1, spq_05 = 0)
spq_score(df, key = spq_key)
```

---

sweat_markers *Calculate sweat-based ionic & metabolic markers*

Description

Computes:

- sweat_chloride (mmol/L)
- Na_K_ratio (sweat Na+/K+)
- sweat_lactate (mmol/L)
- sweat_rate (L/m^2/h) from body mass loss per hour per m^2

Usage

```
sweat_markers(
  data,
  col_map = NULL,
  verbose = TRUE,
  na_action = c("keep", "omit", "error"),
  na_warn_prop = 0.2
)
```

Arguments

data	A data.frame or tibble containing sweat assay and anthropometrics.
col_map	Named list mapping required inputs (defaults assume same names): • sweat_chloride, sweat_Na, sweat_K, sweat_lactate, weight_before, weight_after, duration, body_surface_area
verbose	Logical; if TRUE, prints progress messages and a completion summary.
na_action	One of c("keep","omit","error") for handling missing values in required inputs. Default "keep".
na_warn_prop	Proportion $[0, 1]$ to trigger high-missingness diagnostics for required inputs (debug level). Default 0.2.

Details

Inputs are validated, missingness handled via `na_action`, safe divisions are used to avoid Inf/NaN, and an optional extremes scan/cap is available.

Expected units:

- sweat_chloride, sweat_Na, sweat_K: mmol/L
- sweat_lactate: mmol/L
- weight_before, weight_after: kg
- duration: hours
- body_surface_area: m^2

Value

A tibble with columns: `sweat_chloride`, `Na_K_ratio`, `sweat_lactate`, `sweat_rate`

Note

`sweat_chloride` and `sweat_lactate` are **pass-through** columns; no formula is applied. `Na_K_ratio` is a simple Na/K division. `sweat_rate` uses the mass-loss method: (weight_before - weight_after) / duration / body_surface_area (units: L/m^2/h; assumes 1 kg $\approx$ 1 L). Dill & Costill 1974 describes haematocrit-based blood-volume change, not sweat rate directly; cited here as background context only.

References

Gibson LE, Cooke RE (1959). "A test for concentration of electrolytes in sweat in cystic fibrosis of the pancreas utilizing pilocarpine by iontophoresis." *Pediatrics*, **23**(3), 545–549. (pilocarpine sweat chloride test origin; background) Dill DB, Costill DL (1974). "Calculation of percentage changes in volumes of blood, plasma, and red cells in dehydration." *Journal of Applied Physiology*, **37**(2), 247–248. doi:10.1152/jappl.1974.37.2.247. (dehydration and fluid loss context; background) Farrell PM, White TB, Ren CL, Hempstead SE, Accurso F, Derichs N, Howenstine M, McColley SA, Rock M, Rosenfeld M, Sermet-Gaudelus I, Southern KW, Marshall BC, Sosnay PR (2017). "Diagnosis of cystic fibrosis: consensus guidelines from the Cystic Fibrosis Foundation." *Journal of Pediatrics*, **181S**, S4–S15.e1. doi:10.1016/j.jpeds.2016.09.064. (CF diagnostic sweat chloride cutoffs; background) Sawka MN, Cheuvront SN, Kenefick RW (2015). "Hypohydration and human performance: impact of environment and physiological mechanisms." *Sports Medicine*, **45**(Suppl 1), S51–S60. doi:10.1007/s4027901503957. (sweat rate and hypohydration context; background)

Examples

```
df <- tibble::tibble(
  sweat_chloride    = 45,
  sweat_Na          = 55,
  sweat_K           = 5,
  sweat_lactate     = 4.8,
  weight_before     = 70.0,
  weight_after      = 69.5,
  duration          = 1.0,
```

```
  body_surface_area = 1.9
)
sweat_markers(df)
```

`tracer_dxa_is` *Compute tracer/DXA-based insulin sensitivity indices*

Description

Uses stable isotope tracer infusion rates and DXA-measured fat mass to compute peripheral and adipose insulin sensitivity and related metrics.

Usage

```
tracer_dxa_is(
  data,
  col_map = NULL,
  normalize = NULL,
  na_action = c("keep", "omit", "error"),
  na_warn_prop = 0.2,
  verbose = TRUE
)
```

Arguments

`data`	A data.frame or tibble containing raw measurements.
`col_map`	Named list with entries (depending on mode): Adipose-only required: - I0: fasting insulin (pmol/L) - rate_glycerol, rate_palmitate: tracer rates (mumol/min) - fat_mass, weight, bmi: body composition - HDL_c: HDL cholesterol (mmol/L) Full mode additionally requires: - G0, G30, G120: glucose (mmol/L) - I30, I120: insulin (pmol/L) - TG: triglycerides (mmol/L) - FFA: free fatty acids (mmol/L)
`normalize`	Ignored (kept for backward compatibility).
`na_action`	One of c("keep","omit","error") for NA handling on required inputs. Default "keep".
`na_warn_prop`	Proportion $[0, 1]$ to trigger high-missingness warnings on required inputs. Default 0.2.
`verbose`	Logical; if TRUE, prints progress messages and a completion summary.

Details

Modes:

- Adipose-only indices when only adipose-related keys are mapped (no OGTT glucose/insulin time series)

- Full indices otherwise

Expected units:

- Glucose: mmol/L (internally converted to mg/dL when needed)
- Insulin: pmol/L (internally converted to muU/mL via /6)
- TG: mmol/L (to mg/dL via *88.57); HDL-c: mmol/L (to mg/dL via *38.67)
- Tracer rates: mumol/min
- Fat mass, weight: kg; BMI: kg/m^2

Value

- Adipose-only tibble columns: LIRI_inv, Lipo_inv, ATIRI_inv
- Full-mode tibble columns: I_AUC, FFA_AUC, tracer_palmitate_SI, tracer_glycerol_SI, LIRI_inv, Lipo_inv, ATIRI_inv

Note

`tracer_palmitate_SI` and `tracer_glycerol_SI` are simple rate/fat_mass ratios; the Steele (1959) non-steady-state tracer equation is **not** implemented here. The LIRI formula coefficients (-0.091, 0.4, 0.346, -0.408, 0.435) are attributed to Gastaldelli et al. but the paper cited (`gastaldelli2004betacell`) covers beta-cell dysfunction, not LIRI derivation; the primary LIRI source should be verified. In adipose-only mode (I30 absent) the mean-insulin term uses I0 twice as a fallback.

References

Groop LC, Bonadonna RC, Simonson DC, et al. (1989). "Different Effects of Insulin and Oral Hypoglycemic Agents on Glucose and Lipid Metabolism in Type II Diabetes." *Journal of Clinical Investigation*, **84**(2), 578–585. doi:10.1172/JCI114142. (tracer lipolysis methodology; background) Steele R (1959). "Influences of Glucose Loading and of Injected Insulin on Hepatic Glucose Output." *Annals of the New York Academy of Sciences*, **82**(2), 420–430. doi:10.1111/j.1749-6632.1959.tb44923.x. (tracer dilution theory; Steele equation not directly implemented — background) Roden M, Price TB, Perseghin G, et al. (1996). "Mechanism of Free Fatty Acid-Induced Insulin Resistance in Humans." *Journal of Clinical Investigation*, **97**(12), 2859–2865. doi:10.1172/JCI118742. (FFA-induced insulin resistance mechanism; background) Gastaldelli A, Ferrannini E, Miyazaki Y, Matsuda M, DeFronzo RA (2004). "Beta-Cell Dysfunction and Glucose Intolerance: Results from the San Antonio Metabolism Study." *Diabetologia*, **47**(1), 31–39. doi:10.1007/s0012500312639. (beta-cell dysfunction context; LIRI formula source unverified — background) Karpe F, Dickmann JR, Frayn KN (2011). "Fatty Acids, Obesity, and Insulin Resistance: Time for a Reevaluation." *Diabetes*, **60**(10), 2441–2449. doi:10.2337/db110425. (FFA and insulin resistance review; background) Petersen KF, Dufour S, Savage DB, et al. (2007). "The Role of Skeletal Muscle Insulin Resistance in the Pathogenesis of the Metabolic Syndrome." *Proceedings of the National Academy of Sciences*, **104**(31), 12587–12594. doi:10.1073/pnas.0705408104. (muscle insulin resistance and metabolic syndrome; background) Santomauro AT, Boden G, Silva ME, et al. (1999). "Overnight Lowering of Free Fatty Acids with Acipimox Improves Insulin Resistance and Glucose Tolerance in Obese Diabetic and Nondiabetic Subjects." *Diabetes*, **48**(9), 1836–1841. doi:10.2337/diabetes.48.9.1836. (FFA lowering and insulin sensitivity; background)

Examples

```
df <- data.frame(
  I0 = c(60, 75), rate_glycerol = c(2.1, 2.8), rate_palmitate = c(1.8, 2.3),
  fat_mass = c(18, 24), weight = c(72, 85), BMI = c(24, 29),
  HDL_c = c(1.3, 1.1)
)
col_map <- list(I0="I0", rate_glycerol="rate_glycerol",
                rate_palmitate="rate_palmitate", fat_mass="fat_mass",
                weight="weight", bmi="BMI", HDL_c="HDL_c")
tracer_dxa_is(df, col_map = col_map)
```

---

urine_markers *Calculate urine-only renal and tubular markers (research-ready)*

Description

Computes (urine-only):

- UACR (Albumin-to-Creatinine Ratio, mg/g)
- albuminuria_stage (KDIGO A1/A2/A3 by UACR)
- microalbuminuria flag ("normal" vs "micro")
- UPCR (Urine Protein-to-Creatinine Ratio, mg/g; if urine_protein available)
- U_Na_K_ratio (urine Na+/K+; if urine_Na and urine_K available)
- Creatinine-normalized tubular markers (if present, per g creatinine): NGAL_per_gCr, KIM1_per_gCr, NAG_per_gCr, Beta2Micro_per_gCr, A1Micro_per_gCr, IL18_per_gCr, L_FABP_per_gCr

Usage

```
urine_markers(
  data,
  col_map = NULL,
  verbose = TRUE,
  na_action = c("keep", "omit", "error"),
  na_warn_prop = 0.2
)
```

Arguments

data	A data.frame or tibble with at least urine_albumin and urine_creatinine.
col_map	Optional named list mapping canonical keys (e.g., urine_albumin, urine_creatinine) to actual column names in data. If NULL, column names are inferred automatically.
verbose	Logical; if TRUE, prints progress messages and a completion summary. Default FALSE.

na_action	One of c("keep","omit","error") for handling missing values in required inputs. Default "keep".
na_warn_prop	Proportion $[0, 1]$ to trigger high-missingness warnings for required inputs. Default 0.2.

Details

Inputs are validated, missingness handled via na_action, divisions are safeguarded (Inf/NaN -> NA) with a consolidated zero-denominator warning, and an optional extremes scan/cap is available.

Expected units:

- urine_albumin: mg/L
- urine_protein: mg/L (optional)
- urine_creatinine: mg/dL
- urine_Na, urine_K: mmol/L (optional)
- Optional tubular markers above assumed mg/L when normalized per g creatinine

Value

A tibble with columns: UACR, albuminuria_stage, microalbuminuria, UPCR, U_Na_K_ratio, NGAL_per_gCr, KIM1_per_gCr, NAG_per_gCr, Beta2Micro_per_gCr, A1Micro_per_gCr, IL18_per_gCr, L_FABP_per_gCr

Note

UACR formula: albumin (mg/L) / creatinine (g/L) = albumin (mg/L) $\times$ 100 / creatinine (mg/dL). UPCR and per-gCr tubular markers use the same creatinine denominator: gCr_den = creatinine (mg/dL) \times 0.01 (= g/L). Tubular markers (NGAL, KIM-1, NAG, Beta-2-microglobulin, alpha-1-microglobulin, IL-18, L-FABP) are **pass-through** columns normalised per g creatinine; no formula other than creatinine adjustment is applied.

References

Mogensen CE (1984). "Microalbuminuria predicts clinical proteinuria and early mortality in maturity-onset diabetes." *New England Journal of Medicine*, **310**(6), 356–360. doi:10.1056/NEJM198402093100602. Ginsberg JM, Chang BS, Matarese RA, Garella S (1983). "Use of single voided urine samples to estimate quantitative proteinuria." *New England Journal of Medicine*, **309**(25), 1543–1546. doi:10.1056/NEJM198312223092503. Kidney Disease: Improving Global Outcomes (KDIGO) CKD Work Group (2013). "KDIGO 2012 Clinical Practice Guideline for the Evaluation and Management of Chronic Kidney Disease." *Kidney International Supplements*, **3**(1), 1–150. doi:10.1038/kisup.2012.73, Related synopsis: Stevens and Levin (2013), Ann Intern Med, doi:10.7326/0003-4819-158-11-201306040-00007, https://kdigo.org/guidelines/ckd-evaluation-and-management/. (albuminuria staging A1–A3 UACR cutoffs) de Zeeuw D, Parving H, Henning RH (2006). "Microalbuminuria as an early marker for cardiovascular disease." *Journal of the American Society of Nephrology*, **17**(8), 2100–2105. doi:10.1681/ASN.2006040388. (prognostic UACR validation; background) Ichimura T, Hung CC, Yang SA, Stevens JL, Bonventre JV (2004). "Kidney injury molecule-1: a tissue and urinary biomarker for nephrotoxicant-induced renal injury." *American Journal of Physiology: Renal Physiology*, **286**(3), F552–F563. doi:10.1152/ajprenal.00285.2002. (KIM-1 tubular biomarker; pass-through normalization, background) Portilla D, Dent C, Sugaya T,

others (2008). "Urinary liver-type fatty acid-binding protein as a biomarker of acute kidney injury." *Kidney International*, **73**(4), 465–472. doi:10.1038/sj.ki.5002721. (L-FABP tubular biomarker; pass-through normalization, background)

Examples

```
df <- tibble::tibble(
  urine_albumin    = 30,
  urine_creatinine = 1.2,
  serum_creatinine = 0.9,
  plasma_Na        = 140,
  urine_Na         = 100,
  age              = 55,
  sex              = 2,
  urine_protein    = 150
)
urine_markers(df)
```

---

validate_inputs *Validate required inputs for a calling function*

Description

Ensures required keys exist in `col_map` and have non-empty mappings. Missing keys are reported in a stable order aligned with tests.

Usage

```
validate_inputs(data, col_map, fun_name, required_keys = NULL)
```

Arguments

`data`	data.frame or tibble
`col_map`	named list mapping keys to column names
`fun_name`	character scalar naming the calling function (e.g., "lipid_markers"). Used to look up built-in required keys when `required_keys` is not supplied.
`required_keys`	optional character vector of required col_map keys. When supplied, this takes precedence over the `fun_name` built-in lookup, making the function useful for any caller regardless of `fun_name`.

Value

invisibly TRUE on success; otherwise aborts

Examples

```
df <- data.frame(TG = c(1.5, 2.0), HDL_c = c(1.2, 1.0),
  LDL_c = c(2.0, 2.5), TC = c(4.5, 5.0))
# Using built-in lookup
validate_inputs(df,
  list(TG = "TG", HDL_c = "HDL_c", LDL_c = "LDL_c", TC = "TC"),
  fun_name = "lipid_markers")
# Using explicit required_keys (works for any function)
validate_inputs(df,
  list(TG = "TG", HDL_c = "HDL_c"),
  fun_name = "my_function",
  required_keys = c("TG", "HDL_c"))
```

---

vitamin_d_status *Vitamin D Status Category*

Description

Categorizes vitamin D status based on serum 25-hydroxyvitamin D (25(OH)D) levels.

Usage

```
vitamin_d_status(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error", "ignore", "warn"),
  verbose = TRUE
)
```

Arguments

data	A data.frame or tibble with a 25-hydroxyvitamin D concentration column.
col_map	A named list with vitamin_d giving the column name in data for 25(OH)D.
na_action	One of c("keep","omit","error","ignore","warn").
	• keep/ignore: compute and propagate NA • omit: drop rows with NA in required input • error: abort if required input contains NA • warn: like keep, but emit missingness warnings
verbose	Logical; if TRUE, emits progress via rlang::inform.

Details

Serum 25(OH)D is the standard biomarker for vitamin D status. This function classifies levels (assumed in ng/mL) into categories:

- Deficient (< 20 ng/mL)

- Insufficient (20-29 ng/mL)
- Sufficient (>= 30 ng/mL)

Note: Ensure input units are ng/mL. If values appear extremely high (e.g., median > 150), they might be in nmol/L (divide by 2.5 to convert to ng/mL).

Value

A tibble with one column: vitamin_d_status (ordered factor with levels "Deficient","Insufficient","Sufficient").

References

for Vitamin D IoM(CtRDRI, Calcium (2011). *Dietary Reference Intakes for Calcium and Vitamin D*. National Academies Press. doi:10.17226/13050. Holick MF, Binkley NC, Bischoff-Ferrari HA, et al. (2011). "Evaluation, Treatment, and Prevention of Vitamin D Deficiency: an Endocrine Society Clinical Practice Guideline." *Journal of Clinical Endocrinology & Metabolism*, **96**(7), 1911–1930. doi:10.1210/jc.20110385.

Examples

```
df <- data.frame(VitD = c(18, 45, 72))
vitamin_d_status(df)
```

`vitamin_markers` *Compute composite vitamin and endocrine marker ratios and z-scores*

Description

Given serum/plasma vitamins and related analytes, `vitamin_markers()` computes:

- VitD_Z: z-score of 25(OH)D using provided reference mean/sd
- B12_Fol_Ratio: vitamin B12 / folate
- Ferr_TSat_R: ferritin / transferrin saturation (TSat)
- Cort_DHEA_R: cortisol / DHEA-S
- T_E2_Ratio: testosterone / estradiol
- TSH_fT4_R: TSH / free T4
- Retinol_Z: z-score of retinol using provided reference mean/sd
- Toco_Lip_R: alpha-tocopherol / total lipids
- Mg_Zn_R: magnesium / zinc
- Cu_Zn_R: copper / zinc Plus pass-through: PIVKA_II, VitC, Homocysteine, MMA

Usage

```
vitamin_markers(
  data,
  col_map = NULL,
  na_action = c("keep", "omit", "error"),
  na_warn_prop = 0.2,
  verbose = TRUE
)
```

Arguments

data	A data.frame or tibble with vitamin/analyte columns.
col_map	Named list mapping required keys to column names: VitD, VitD_ref_mean, VitD_ref_sd, B12, Folate, Ferritin, TSat, Cortisol, DHEAS, Testosterone, Estradiol, TSH, free_T4, Retinol, Retinol_ref_mean, Retinol_ref_sd, Tocopherol, Total_lipids, PIVKA_II, VitC, Homocysteine, MMA, Magnesium, Zinc, Copper.
na_action	One of c("keep","omit","error") for required inputs. Default "keep".
na_warn_prop	Proportion $[0, 1]$ to trigger high-missingness debug notices. Default 0.2.
verbose	Logical; if TRUE (default), prints column mapping, input availability, physiological range information (informational only, values not altered), the list of markers being computed with their inputs, and a per-column results summary.

Details

HM-CS v2:

- Validation via `hm_validate_inputs(data, col_map, required_keys, fn)`
- User errors via `rlang::abort(..., class=...)`
- Verbosity via `hm_inform(level)` controlled by `options(healthmarkers.verbose)`
- High-missingness diagnostics at debug level only

Value

A tibble with columns: VitD_Z, B12_Fol_Ratio, Ferr_TSat_R, Cort_DHEA_R, T_E2_Ratio, TSH_fT4_R, Retinol_Z, Toco_Lip_R, PIVKA_II, VitC, Homocysteine, MMA, Mg_Zn_R, Cu_Zn_R. If an ID column is detected in `data` (e.g. `id`, `IID`, `participant_id`), it is prepended as the first output column.

Note

`VitD_Z` and `Retinol_Z` are z-scores using **user-supplied** reference mean and SD; no population reference equations are applied. All ratio markers (`B12_Fol_Ratio`, `Ferr_TSat_R`, `Cort_DHEA_R`, `T_E2_Ratio`, `TSH_fT4_R`, `Toco_Lip_R`, `Mg_Zn_R`, `Cu_Zn_R`) are simple numerator/denominator divisions. `PIVKA_II`, `VitC`, `Homocysteine`, and `MMA` are **pass-through** columns; no formula is applied.

References

Holick MF (2007). "Vitamin D Deficiency." *New England Journal of Medicine*, **357**(3), 266–281. doi:10.1056/NEJMra070553. (vitamin D deficiency review; background) O'Leary F, Samman S (2010). "Vitamin B12 in Health and Disease." *Nutrients*, **2**(3), 299–316. doi:10.3390/nu2030299. (vitamin B12 in health and disease; background) Ganz T, Nemeth E (2015). "Iron homeostasis in host defence and inflammation." *Nature Reviews Immunology*, **15**(8), 500–510. doi:10.1038/nri3863. (iron homeostasis and ferritin; background)

Examples

```
# All 25 required columns must be supplied
df <- data.frame(
  VitD = 50, VitD_ref_mean = 40, VitD_ref_sd = 5,
  B12 = 300, Folate = 15, Ferritin = 80, TSat = 0.25,
  Cortisol = 200, DHEAS = 100, Testosterone = 12, Estradiol = 120,
  TSH = 2, free_T4 = 14, Retinol = 0.8, Retinol_ref_mean = 0.9,
  Retinol_ref_sd = 0.2, Tocopherol = 30, Total_lipids = 3,
  PIVKA_II = 5, VitC = 60, Homocysteine = 10, MMA = 0.3,
  Magnesium = 0.8, Zinc = 15, Copper = 15
)
vitamin_markers(df, verbose = FALSE)
```

---

who5_score *WHO-5 scoring*

Description

WHO-5 scoring

Usage

```
who5_score(
  data,
  col_map = list(),
  na_action = c("keep", "omit", "error"),
  missing_prop_max = 0.2,
  impute = c("none", "mean"),
  prefix = "WHO5",
  low_cutoff_percent = 50,
  verbose = TRUE
)
```

Arguments

`data`	Data frame containing questionnaire item columns.
`col_map`	Named list mapping canonical item IDs to column names; defaults assume items are already named.
`na_action`	How to handle rows with missing items: `keep`, `omit`, or `error`.
`missing_prop_max`	Maximum allowed proportion of missing items per row before the score is set to `NA`.
`impute`	Imputation strategy for missing items when under the threshold: `none` or `mean` (row-wise mean).
`prefix`	Prefix for output column names.
`low_cutoff_percent`	Percentage threshold for low well-being flag.
`verbose`	Logical; if `TRUE`, emits informational messages about column resolution and scoring progress via `hm_inform()`.

Value

A tibble of score columns only: `WHO5_raw`, `WHO5_percent`, `WHO5_low_wellbeing`. Input columns are not included.

References

Topp CW, Østergaard SrD, Sø ndergaard S, Bech P (2015). "The WHO-5 Well-Being Index: A Systematic Review of the Literature." *Psychotherapy and Psychosomatics*, **84**(3), 167–176. doi:10.1159/000376585.

Examples

```
df <- data.frame(who5_01 = 0, who5_02 = 1, who5_03 = 2, who5_04 = 3, who5_05 = 4)
who5_score(df)
```

Index

∗ **respiratory-markers**
    spirometry_markers, 106

adipo_is, 4
adiposity_sds, 6, *13*
adiposity_sds_strat, 8, *13*
all_health_markers, 9
all_insulin_indices, 11
allostatic_load, 12
alm_bmi_index, 14
asrs_score, 15
atherogenic_indices, 17

bis_score, 18
bode_index, 19
bone_markers, 20

calc_sds, 22
charlson_index, 24
ckd_stage, 26
cognitive_score, 27
corrected_calcium, 28
cvd_marker_aip, 29
cvd_marker_ldl_particle_number, 30
cvd_risk, 31
cvd_risk_ascvd, 32
cvd_risk_qrisk3, 33
cvd_risk_scorescvd, 34
cvd_risk_stroke, 35

fasting_is, 36
fasting_is(), *91*
frailty_index, 37
frax_score, 40

gad7_score, 41
ghq12_score, 42
glycemic_markers, 43
glycemic_markers(), *53*

health_summary, 46
hm_col_report, 47
hm_col_report(), *9*
hm_normalize, 48
hormone_markers, 49

iAge, 51
iAge(), *64*, *72*
impute_mice, 53
impute_missforest, 55
impute_missing, 56
impute_missing(), *53*, *64*
inflammatory_markers, 57
inflammatory_markers(), *64*, *72*, *78*
isi_score, 59

k10_score, 60
k6_score, 61
kidney_failure_risk, 63
kidney_failure_risk(), *72*, *78*
kyn_trp_ratio, 65

lipid_markers, 66
lipid_markers(), *78*
liver_fat_markers, 68
liver_markers, 70
liver_markers(), *78*

marker_summary, 73
mdq_score, 73
metabolic_markers, 75
metabolic_risk_features, 76
metss, 78

nfl_marker, 80
normalize_vec, 82
normalize_vec(), *48*, *49*, *91*
nutrient_markers, 83

obesity_indices, 86
ogtt_is, 89
oxidative_markers, 92

phq9_score, 93
plot_frailty_age, 94
psych_dx_flags, 95
psych_markers, 96
psych_med_flags, 97
pulmo_markers, 98

renal_markers, 100

saliva_markers, 102
sarc_f_score, 105
spirometry_markers, 106
spq_score, 107
sweat_markers, 108

tracer_dxa_is, 110

urine_markers, 112

validate_inputs, 114
vitamin_d_status, 115
vitamin_markers, 116

who5_score, 118