

# Package: PhysioClinical (via r-universe)

October 3, 2026

**Type** Package

**Title** Clinical Outcome Measures and Responder Analysis

**Version** 0.7.0

**Description** Validated clinical outcome-measure scoring (FMA, Berg, ARAT, TUG, 6MWT, Barthel, Katz, Lawton-Brody IADL, ...), MDC/MCID responder classification, ICF tagging, interval (Rasch) analysis of ordinal ADL/IADL instruments, and normative deviation scoring for the x-biosignal rehabilitation ecosystem.

**Depends** R (>= 4.2)

**Imports** PhysioExperiment (>= 2.0.0), methods, stats, utils, yaml

**Suggests** PhysioCore, testthat (>= 3.2.0), jsonlite, jsonvalidate, PhysioAppKit, PhysioMoCap, PhysioIO, PhysioAnnotationHub, knitr, rmarkdown

**License** MIT + file LICENSE

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.3

**VignetteBuilder** knitr

**URL** <https://github.com/x-biosignal/PhysioClinical>,  
<https://x-biosignal.github.io/PhysioClinical/>

**BugReports** <https://github.com/x-biosignal/PhysioExperiment/issues>

**Config/testthat/edition** 3

**Config/pak/sysreqs** zlib1g-dev

**Repository** <https://x-biosignal.r-universe.dev>

**Date/Publication** 2026-10-03 13:11:47 UTC

**RemoteUrl** <https://github.com/x-biosignal/PhysioClinical>

**RemoteRef** HEAD

**RemoteSha** 5a301f33e90ac70c7b55c770ca6eab08bf3acc2b

## Contents

|                                            |    |
|--------------------------------------------|----|
| applyEquating . . . . .                    | 3  |
| as.data.frame.ClinicalScore . . . . .      | 4  |
| classifyResponder . . . . .                | 5  |
| ClinicalInstrument . . . . .               | 6  |
| ClinicalInstrument-class . . . . .         | 7  |
| ClinicalScore-class . . . . .              | 8  |
| defineGoal . . . . .                       | 8  |
| equateScores . . . . .                     | 10 |
| estimateMCID_anchor . . . . .              | 11 |
| estimateMCID_distribution . . . . .        | 12 |
| estimateMDC . . . . .                      | 13 |
| fromFHIR . . . . .                         | 14 |
| gasSummary . . . . .                       | 15 |
| getClinimetric . . . . .                   | 15 |
| getInstrument . . . . .                    | 16 |
| getNormative . . . . .                     | 17 |
| GovernedNormativeReference . . . . .       | 17 |
| GovernedNormativeReference-class . . . . . | 18 |
| listClinimetrics . . . . .                 | 19 |
| listInstruments . . . . .                  | 20 |
| listNormative . . . . .                    | 20 |
| masLevelLabel . . . . .                    | 21 |
| matchStratum . . . . .                     | 21 |
| mdcResponder . . . . .                     | 22 |
| normativeDeviation . . . . .               | 23 |
| normativeZScore . . . . .                  | 23 |
| promisRawToT . . . . .                     | 24 |
| raschAnalyze . . . . .                     | 25 |
| raschRecode . . . . .                      | 26 |
| raschResponder . . . . .                   | 27 |
| registerInstrument . . . . .               | 28 |
| registerNormative . . . . .                | 29 |
| romNormalcy . . . . .                      | 29 |
| romReference . . . . .                     | 30 |
| score10MWT . . . . .                       | 31 |
| score6MWT . . . . .                        | 32 |
| scoreARAT . . . . .                        | 33 |
| scoreBarthel . . . . .                     | 34 |
| scoreBerg . . . . .                        | 34 |
| scoreBPI . . . . .                         | 35 |
| scoreCohort . . . . .                      | 36 |
| scoreEQ5D5L . . . . .                      | 37 |
| scoreFAM . . . . .                         | 37 |
| scoreFIM . . . . .                         | 38 |
| scoreFMALE . . . . .                       | 39 |
| scoreFMAUE . . . . .                       | 39 |

|                                                  |           |
|--------------------------------------------------|-----------|
| scoreFSS . . . . .                               | 40        |
| scoreGAS . . . . .                               | 41        |
| scoreInstrument . . . . .                        | 42        |
| scoreKatz . . . . .                              | 43        |
| scoreLawton . . . . .                            | 43        |
| scoreMAS . . . . .                               | 44        |
| scoreMFIS . . . . .                              | 45        |
| scoreMMSE . . . . .                              | 45        |
| scoreMoCA . . . . .                              | 46        |
| scoreMRCSum . . . . .                            | 47        |
| scoreMRS . . . . .                               | 48        |
| scoreNIHSS . . . . .                             | 48        |
| scorePainNRS . . . . .                           | 49        |
| scorePROMIS . . . . .                            | 50        |
| scoreSF36 . . . . .                              | 51        |
| scoreTUG . . . . .                               | 52        |
| scoreWMFT . . . . .                              | 53        |
| show,GovernedNormativeReference-method . . . . . | 53        |
| toADaM_ADQS . . . . .                            | 54        |
| toCDISC_QS . . . . .                             | 54        |
| toFHIRObservation . . . . .                      | 56        |
| toOMOP . . . . .                                 | 57        |
| validateFHIRObservation . . . . .                | 58        |
| validateNormativeManifest . . . . .              | 58        |
| wmftMedianTime . . . . .                         | 59        |
| writeFHIRBundle . . . . .                        | 60        |
| writeOMOPTables . . . . .                        | 60        |
| <b>Index</b>                                     | <b>61</b> |

---

|               |                                              |
|---------------|----------------------------------------------|
| applyEquating | <i>Apply a score crosswalk to new values</i> |
|---------------|----------------------------------------------|

---

**Description**

Apply a score crosswalk to new values

**Usage**

applyEquating(equating, values)

**Arguments**

- |          |                                                        |
|----------|--------------------------------------------------------|
| equating | a score_equating from <a href="#">equateScores()</a> . |
| values   | numeric source-scale values to convert.                |

**Value**

numeric target-scale equivalents (linearly interpolated within the crosswalk, clamped to its range).

**See Also**

[equateScores\(\)](#)

**Examples**

```
set.seed(1)
ability <- rnorm(200)
barthel <- pmin(100, pmax(0, round((ability + 3) * 16 / 5) * 5))
fim <- pmin(126, pmax(18, round(18 + (ability + 3) * 18)))
applyEquating(equateScores(barthel, fim), c(40, 60, 80))
```

---

as.data.frame.ClinicalScore

*Convert a ClinicalScore to a one-row data.frame*

---

**Description**

Convert a ClinicalScore to a one-row data.frame

**Usage**

```
## S3 method for class 'ClinicalScore'
as.data.frame(x, ..., stringsAsFactors = FALSE)
```

**Arguments**

|                  |                                              |
|------------------|----------------------------------------------|
| x                | A <a href="#">ClinicalScore</a> .            |
| ...              | Ignored.                                     |
| stringsAsFactors | Ignored (kept for the S3 generic signature). |

**Value**

A one-row data.frame with the total, each subscale, the stratum and the metadata columns.

---

|                   |                                                  |
|-------------------|--------------------------------------------------|
| classifyResponder | <i>Dual MDC-vs-MCID responder classification</i> |
|-------------------|--------------------------------------------------|

---

## Description

Classifies each subject's pre-to-post change with the two-criterion responder framework (Beaton 2001; de Vet 2006): a change is cross-tabulated on whether it exceeds the Minimal Detectable Change (is it real, above measurement error?) and the Minimal Clinically Important Difference (is it clinically meaningful?), giving four categories — `true_responder` (exceeds both), `subclinical_change` (real but below MCID), `measurement_error` (claims MCID but within noise, only when  $MCID < MDC$ ) and `non_responder`. Vectorized over subjects and direction-aware.

## Usage

```
classifyResponder(
  baseline,
  followup,
  instrument = NULL,
  population = NULL,
  mdc = NULL,
  mcid = NULL,
  direction = c("increase", "decrease")
)

## S3 method for class 'responder_classification'
summary(object, ...)
```

## Arguments

|                                     |                                                                                                               |
|-------------------------------------|---------------------------------------------------------------------------------------------------------------|
| <code>baseline, followup</code>     | Numeric vectors of pre / post scores (one per subject).                                                       |
| <code>instrument, population</code> | Optional instrument / population used to look up mdc/mcid from the clinimetric store when they are not given. |
| <code>mdc, mcid</code>              | Optional explicit MDC / MCID thresholds (positive); override the store lookup.                                |
| <code>direction</code>              | "increase" (default) if higher is better, or "decrease" if lower is better.                                   |
| <code>object</code>                 | A "responder_classification".                                                                                 |
| <code>...</code>                    | Unused.                                                                                                       |

## Value

A "responder\_classification" data.frame with the change, the direction-aware improvement, the MDC/MCID crossing flags and the four-level classification factor. A subject with a missing baseline/followup gets NA flags and classification (missing, not assumed a non-responder); `summary()` counts it.

`summary` returns the MDC x MCID contingency table.

## References

Beaton DE et al. (2001); de Vet HCW et al. (2006).

## See Also

`estimateMDC()`, `getClinimetric()`, `mdcResponder()`

## Examples

```
classifyResponder(c(20, 25, 30), c(31, 26, 45), mdc = 5.2, mcid = 9)
```

---

|                    |                                       |
|--------------------|---------------------------------------|
| ClinicalInstrument | <i>Construct a ClinicalInstrument</i> |
|--------------------|---------------------------------------|

---

## Description

Construct a ClinicalInstrument

## Usage

```
ClinicalInstrument(
  id,
  name = NA_character_,
  version = NA_character_,
  items,
  item_ranges,
  item_type = "interval",
  subscales = list(),
  aggregation = "sum",
  direction = "higher_better",
  strata = list(),
  item_values = list(),
  item_recode = list(),
  source_ref = NA_character_
)
```

## Arguments

`id`, `name`, `version`, `items`, `item_ranges`, `item_type`, `subscales`, `aggregation`, `direction`, `strata`, `item_values`, `item_recode`, `source_ref`

Instrument specification fields (see the class slots).

## Value

A ClinicalInstrument.

**See Also**

`scoreInstrument()`, `registerInstrument()`

**Examples**

```
ClinicalInstrument(
  id = "toy", name = "Toy scale", items = c("q1", "q2"),
  item_ranges = list(q1 = c(0, 4), q2 = c(0, 4)))
```

---

ClinicalInstrument-class

*Clinical instrument specification*

---

**Description**

A declarative specification of a clinical outcome measure: its items, their admissible ranges and measurement level, its subscales, how items aggregate into scores, the direction of clinical benefit, and the interpretation strata.

**Usage**

```
## S4 method for signature 'ClinicalInstrument'
show(object)
```

**Arguments**

`object` A ClinicalInstrument.

**Slots**

`id` Short unique identifier (e.g. "berg").

`name` Human-readable name.

`version` Specification version string.

`items` Character vector of item identifiers, in order.

`item_ranges` Named list; each entry a length-2 numeric `c(min, max)` for the corresponding item.

`item_type` Character vector (recycled to the item count) of "interval" or "ordinal".

`subscales` Named list mapping each subscale name to a character vector of its item identifiers; length 0 for single-scale instruments.

`aggregation` "sum" or "mean".

`direction` "higher\_better" or "higher\_worse".

`strata` List of interpretation bands, each a list with label, lower and upper.

`item_values` Optional named list; for an item with a discrete (possibly non-integer) admissible value set - e.g. the Modified Ashworth 1+ = 1.5 - the numeric vector of allowed values. Items absent from the list are validated only against `item_ranges` / `item_type`.

item\_recode Optional named list; for an item whose raw responses are recoded before scoring (e.g. the SF-36 0-100 recode, or a reversed item), a named numeric map from the raw response (a character key) to the scored value. Applied after validation, before aggregation.

source\_ref Citation / provenance string.

---

|                     |                       |
|---------------------|-----------------------|
| ClinicalScore-class | Clinical score result |
|---------------------|-----------------------|

---

**Description**

The result of scoring a set of item responses against a [ClinicalInstrument](#).

**Usage**

```
## S4 method for signature 'ClinicalScore'
show(object)
```

**Arguments**

object                    A ClinicalScore.

**Slots**

instrument\_id Identifier of the scoring instrument.

total Total score (or NA if unresolved under the missing policy).

subscales Named numeric vector of subscale scores.

stratum Interpretation stratum label the total falls in (or NA).

items\_used Character vector of the item identifiers actually scored.

missing\_handling The missing-data policy applied.

timestamp Optional character timestamp.

subject\_id Optional subject identifier.

---

|            |                                       |
|------------|---------------------------------------|
| defineGoal | Define a Goal Attainment Scaling goal |
|------------|---------------------------------------|

---

**Description**

Describes one GAS goal: its attainment levels (typically -2:2 with 0 = expected outcome), a weight, and an optional ICF linkage. When both importance and difficulty are given the (Turner-Stokes) weight is their product; otherwise weight is used (default 1, i.e. unweighted GAS).

**Usage**

```
defineGoal(  
  description,  
  levels = -2:2,  
  weight = 1,  
  importance = NULL,  
  difficulty = NULL,  
  icf_tag = NULL  
)
```

**Arguments**

|                        |                                                                                                      |
|------------------------|------------------------------------------------------------------------------------------------------|
| description            | Free-text goal description.                                                                          |
| levels                 | Ordered numeric attainment levels (must include 0, the expected outcome); default -2:2.              |
| weight                 | Goal weight when importance/difficulty are not supplied (default 1, positive).                       |
| importance, difficulty | Optional 0+ ratings; if both given the weight is $\text{importance} * \text{difficulty}$ .           |
| icf_tag                | Optional ICF code (e.g. "d450") linking the goal to the International Classification of Functioning. |

**Value**

A "gas\_goal" object.

**References**

Kiresuk & Sherman (1968); Turner-Stokes (2009).

**See Also**

[scoreGAS\(\)](#)

**Examples**

```
defineGoal("Walk 100 m unaided", importance = 3, difficulty = 2,  
  icf_tag = "d450")
```

---

|              |                                                                       |
|--------------|-----------------------------------------------------------------------|
| equateScores | <i>Equate one clinical score scale to another (build a crosswalk)</i> |
|--------------|-----------------------------------------------------------------------|

---

## Description

Constructs a conversion from a "source" scale to a "target" scale using a linking sample measured on both. Equipercentile equating (the default) maps a source score to the target score with the same percentile rank, so it handles the non-linear, bounded relationship typical of ADL scales; linear equating matches only the mean and SD.

## Usage

```
equateScores(
  from,
  to,
  method = c("equipercentile", "linear"),
  from_scores = sort(unique(from[is.finite(from)]))
)
```

## Arguments

|             |                                                                                                                                            |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| from        | Numeric source-scale scores from the linking sample.                                                                                       |
| to          | Numeric target-scale scores from the linking sample (equipercentile uses the two marginal distributions; linear uses their means and SDs). |
| method      | "equipercentile" (default) or "linear".                                                                                                    |
| from_scores | Source scores to tabulate (default: the sorted unique observed source scores).                                                             |

## Value

an object of class `score_equating`: a list with `table` (data.frame from, to), `method`, and the linking-sample size. Convert new values with [applyEquating\(\)](#).

## References

Kolen MJ, Brennan RL (2004) Test Equating, Scaling, and Linking; Houlden H et al. (2006) Clin Rehabil 20:153-159 (Barthel<->FIM).

## See Also

[applyEquating\(\)](#)

**Examples**

```
set.seed(1)
ability <- rnorm(200)
barthel <- pmin(100, pmax(0, round((ability + 3) * 16 / 5) * 5))
fim <- pmin(126, pmax(18, round(18 + (ability + 3) * 18)))
eq <- equateScores(barthel, fim)          # Barthel -> FIM crosswalk
head(eq$table)
```

---

|                     |                          |
|---------------------|--------------------------|
| estimateMCID_anchor | <i>Anchor-based MCID</i> |
|---------------------|--------------------------|

---

**Description**

Estimates the MCID from change scores anchored to an external improvement indicator. "roc" returns the change cut-off that maximizes the Youden index for classifying the anchor; "mean\_change" the mean change among the (minimally) improved; "predictive" the change at which a logistic model of the anchor crosses probability 0.5.

**Usage**

```
estimateMCID_anchor(
  change,
  anchor,
  method = c("roc", "mean_change", "predictive"),
  direction = c("increase", "decrease")
)
```

**Arguments**

|           |                                                                                                            |
|-----------|------------------------------------------------------------------------------------------------------------|
| change    | Numeric change scores (follow-up minus baseline).                                                          |
| anchor    | Improvement indicator aligned to change: logical, or 0/1, where TRUE/1 marks an (minimally) improved case. |
| method    | "roc" (default), "mean_change" or "predictive".                                                            |
| direction | "increase" (default) if a higher change means improvement, or "decrease" if a lower change does.           |

**Value**

The MCID estimate (on the change scale).

**References**

Jaeschke et al. (1989); Copay et al. (2007).

**See Also**

[estimateMDC\(\)](#), [estimateMCID\\_distribution\(\)](#)

## Examples

```
set.seed(1)
change <- c(rnorm(50, 1, 2), rnorm(50, 8, 2))
anchor <- rep(c(0, 1), each = 50)
estimateMCID_anchor(change, anchor)
```

---

estimateMCID\_distribution

*Distribution-based MCID (fraction-of-SD rule)*

---

## Description

The distribution-based Minimal Clinically Important Difference as a fraction of the baseline standard deviation (Norman et al. 2003: the 0.5 SD rule).

## Usage

```
estimateMCID_distribution(baseline_sd, fraction = 0.5)
```

## Arguments

|             |                                             |
|-------------|---------------------------------------------|
| baseline_sd | Baseline standard deviation of the outcome. |
| fraction    | Fraction of the SD (default 0.5).           |

## Value

The MCID estimate (fraction \* baseline\_sd).

## References

Norman GR, Sloan JA, Wyrwich KW (2003). *Med Care* 41(5).

## See Also

[estimateMDC\(\)](#), [estimateMCID\\_anchor\(\)](#)

## Examples

```
estimateMCID_distribution(baseline_sd = 12)
```

estimateMDC

*Distribution-based Minimal Detectable Change from test-retest data***Description**

Estimates the MDC from a test-retest reliability study: the intraclass correlation gives the reliability, the standard error of measurement is  $SD\sqrt{1 - ICC}$ , and  $MDC = SEM \times z \times \sqrt{2}$  (delegating to `PhysioCore::icc/sem/mdc`).

**Usage**

```
estimateMDC(
  test_retest,
  method = "distribution",
  confidence = 0.95,
  model = c("twoway", "oneway")
)
```

**Arguments**

|                          |                                                                                  |
|--------------------------|----------------------------------------------------------------------------------|
| <code>test_retest</code> | A subjects-by-occasions numeric matrix (or data.frame) of repeated measurements. |
| <code>method</code>      | Estimation method; currently "distribution".                                     |
| <code>confidence</code>  | Confidence level for the MDC (default 0.95).                                     |
| <code>model</code>       | ICC model passed to <code>PhysioCore::icc</code> (default "twoway").             |

**Value**

A list with `mdc`, `sem`, `icc` and `confidence`.

**References**

Shrout & Fleiss (1979); de Vet et al. (2006).

**See Also**

[estimateMCID\\_distribution\(\)](#), [estimateMCID\\_anchor\(\)](#)

**Examples**

```
set.seed(1)
base <- rnorm(40, 50, 10)
retest <- base + rnorm(40, 0, 3)
estimateMDC(cbind(base, retest))
```

fromFHIR

*Import a FHIR R4 Observation as a clinical score*

## Description

Parses an HL7 FHIR R4 Observation (as produced by [toFHIRObservation](#)) back into a [ClinicalScore](#), the import counterpart of the `to*/write*` exporters. The `instrument_id` and each subscale name are recovered from the bundled LOINC map (reverse code lookup, then display, then the verbatim text); the total and subscale values come from `valueQuantity` (a `dataAbsentReason` becomes NA); the subject id strips a leading "Patient/"; and the timestamp comes from `effectiveDateTime`.

## Usage

```
fromFHIR(observation)
```

## Arguments

|             |                                                                                                                                                                      |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| observation | A FHIR Observation as a list (a <a href="#">toFHIRObservation</a> result or a parsed <b>jsonlite</b> object), a JSON string, or a path to a <code>.json</code> file. |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## Details

The round-trip is lossy for three slots that [toFHIRObservation](#) does not serialize: `items_used` and `missing_handling` cannot be recovered, and `stratum` is re-derived from the total only when the instrument is registered (via [getInstrument](#)), otherwise NA.

## Value

A [ClinicalScore](#).

## See Also

[toFHIRObservation](#), [writeFHIRBundle](#)

## Examples

```
sc <- scoreInstrument("katz_adl",
  stats::setNames(rep(1, 6), getInstrument("katz_adl")@items),
  subject_id = "P01")
fromFHIR(toFHIRObservation(sc))@total
```

---

|            |                              |
|------------|------------------------------|
| gasSummary | <i>Tabulate a GAS result</i> |
|------------|------------------------------|

---

**Description**

Tabulate a GAS result

**Usage**

```
gasSummary(object, ...)
```

**Arguments**

|        |                                                |
|--------|------------------------------------------------|
| object | A "gas_result" from <a href="#">scoreGAS</a> . |
| ...    | Unused.                                        |

**Value**

A data.frame with one row per goal (description, ICF tag, weight, attained level and its weighted contribution) plus the T-score as an attribute.

**See Also**

[scoreGAS\(\)](#)

---

|                |                                                      |
|----------------|------------------------------------------------------|
| getClinimetric | <i>Look up a per-instrument clinimetric constant</i> |
|----------------|------------------------------------------------------|

---

**Description**

Retrieves a published clinimetric statistic (MDC, MCID, MCII, SEM, ICC, ...) for a measurement instrument from the packaged, provenanced constants store. The return value carries the source reference (DOI, population n, provenance note) so a looked-up threshold is always attributable.

**Usage**

```
getClinimetric(instrument, statistic, population = NULL)
```

**Arguments**

|            |                                                                                                |
|------------|------------------------------------------------------------------------------------------------|
| instrument | Instrument name (e.g. "FMA-UE", "10mWT"); case insensitive.                                    |
| statistic  | One of "MDC", "MCID_anchor", "MCID_dist", "MCII", "SEM", "ICC" (case insensitive).             |
| population | Optional population/context key to disambiguate multiple rows (e.g. "chronic_stroke_minimal"). |

**Value**

A "clinimetric" data.frame of the matching row(s) with value, CI, method, reference\_doi, population\_n and provenance\_note; or NA with a warning when nothing (or no such population) matches.

**See Also**

[listClinimetrics\(\)](#)

**Examples**

```
getClinimetric("10mWT", "MCII")
getClinimetric("FMA-UE", "MCID_anchor", population = "chronic_stroke_minimal")
```

---

|               |                                                  |
|---------------|--------------------------------------------------|
| getInstrument | <i>Retrieve a registered clinical instrument</i> |
|---------------|--------------------------------------------------|

---

**Description**

Retrieve a registered clinical instrument

**Usage**

```
getInstrument(id)
```

**Arguments**

|    |                        |
|----|------------------------|
| id | Instrument identifier. |
|----|------------------------|

**Value**

The [ClinicalInstrument](#).

**See Also**

[listInstruments\(\)](#), [registerInstrument\(\)](#)

---

|              |                                                         |
|--------------|---------------------------------------------------------|
| getNormative | <i>Retrieve a normative reference from the registry</i> |
|--------------|---------------------------------------------------------|

---

**Description**

Retrieve a normative reference from the registry

**Usage**

```
getNormative(id, version = "latest", root = NULL)
```

**Arguments**

|         |                                                                             |
|---------|-----------------------------------------------------------------------------|
| id      | Artifact identifier.                                                        |
| version | Semantic version, or "latest" (default) for the highest registered version. |
| root    | Registry root (see <a href="#">registerNormative()</a> ).                   |

**Value**

The stored [GovernedNormativeReference](#).

**See Also**

[registerNormative\(\)](#), [listNormative\(\)](#)

---

|                            |                                                 |
|----------------------------|-------------------------------------------------|
| GovernedNormativeReference | <i>Construct a governed normative reference</i> |
|----------------------------|-------------------------------------------------|

---

**Description**

Construct a governed normative reference

**Usage**

```
GovernedNormativeReference(  
  id,  
  modality,  
  metric,  
  version = "1.0.0",  
  provenance = list(),  
  consent = list(),  
  license = list(),  
  governance = list(),  
  strata_vars = character(0),  
  model = list(),  
  n = NA_real_  
)
```

**Arguments**

|                                          |                                                                                                                                         |
|------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| id, modality, metric                     | Identifying strings.                                                                                                                    |
| version                                  | Semantic version (default "1.0.0").                                                                                                     |
| provenance, consent, license, governance | Governance lists (see <a href="#">GovernedNormativeReference</a> ); provenance\$source, consent\$status and license\$spdx are required. |
| strata_vars                              | Character vector of stratification variables.                                                                                           |
| model                                    | List holding the normative model (stratified mean/sd table or LMS coefficients).                                                        |
| n                                        | Reference sample size.                                                                                                                  |

**Value**

A validated [GovernedNormativeReference](#).

**See Also**

[registerNormative\(\)](#), [validateNormativeManifest\(\)](#)

**Examples**

```
GovernedNormativeReference(
  id = "gait_speed_adult", modality = "gait", metric = "gait_speed",
  version = "1.0.0",
  provenance = list(source = "Bohannon 1997", doi = "10.1093/ageing/26.1.15"),
  consent = list(status = "public_aggregate"),
  license = list(spdx = "CC-BY-4.0", redistribution_ok = TRUE),
  governance = list(custodian = "Matsui Lab", access_level = "open"),
  strata_vars = c("age", "sex"),
  model = list(type = "strata",
    table = data.frame(age = 70, sex = "M", mean = 1.3, sd = 0.2)),
  n = 230)
```

---

GovernedNormativeReference-class

*Governed normative reference*

---

**Description**

An S4 container for a normative reference distribution together with the governance metadata a clinical artifact must carry: provenance, consent, license and custodianship. Validity requires the provenance source, consent status and license identifier to be present, so an ungoverned reference cannot be constructed.

**Slots**

**id** Stable artifact identifier.  
**modality** Signal modality (e.g. "gait", "hrv").  
**metric** The normed quantity (e.g. "gait\_speed").  
**version** Semantic version string "x.y.z".  
**provenance** List with at least source; optionally doi, collection\_date.  
**consent** List with at least status; optionally ethics\_id.  
**license** List with at least spdx; optionally redistribution\_ok.  
**governance** List with custodian and access\_level.  
**strata\_vars** Character vector of stratification variables (e.g. c("age", "sex")).  
**model** List holding the normative model — either a stratified mean/sd table or LMS (lambda/mu/sigma) coefficients.  
**n** Reference sample size.

---

|                  |                                             |
|------------------|---------------------------------------------|
| listClinimetrics | <i>List available clinimetric constants</i> |
|------------------|---------------------------------------------|

---

**Description**

List available clinimetric constants

**Usage**

```
listClinimetrics(instrument = NULL)
```

**Arguments**

**instrument** Optional instrument to filter by (case insensitive).

**Value**

A data.frame of the stored clinimetric constants.

**See Also**

[getClinimetric\(\)](#)

**Examples**

```
listClinimetrics("6MWT")
```

---

|                 |                                            |
|-----------------|--------------------------------------------|
| listInstruments | <i>List available clinical instruments</i> |
|-----------------|--------------------------------------------|

---

**Description**

List available clinical instruments

**Usage**

```
listInstruments()
```

**Value**

A character vector of registered instrument ids.

**See Also**

[getInstrument\(\)](#), [registerInstrument\(\)](#)

---

|               |                                             |
|---------------|---------------------------------------------|
| listNormative | <i>List registered normative references</i> |
|---------------|---------------------------------------------|

---

**Description**

List registered normative references

**Usage**

```
listNormative(root = NULL)
```

**Arguments**

|      |                                                           |
|------|-----------------------------------------------------------|
| root | Registry root (see <a href="#">registerNormative()</a> ). |
|------|-----------------------------------------------------------|

**Value**

A data.frame of id, version, modality, metric, access\_level and n, newest version last.

**See Also**

[registerNormative\(\)](#), [getNormative\(\)](#)

---

|               |                                                        |
|---------------|--------------------------------------------------------|
| masLevelLabel | <i>Modified Ashworth Scale level label for a value</i> |
|---------------|--------------------------------------------------------|

---

**Description**

The inverse of the MAS arithmetic mapping: returns the ordinal label for a numeric MAS value (so 1.5 round-trips to "1+").

**Usage**

```
masLevelLabel(value)
```

**Arguments**

|       |                       |
|-------|-----------------------|
| value | Numeric MAS value(s). |
|-------|-----------------------|

**Value**

Character MAS level label(s).

**See Also**

[scoreMAS\(\)](#)

**Examples**

```
masLevelLabel(1.5)
```

---

|              |                                                            |
|--------------|------------------------------------------------------------|
| matchStratum | <i>Match the normative stratum for a set of covariates</i> |
|--------------|------------------------------------------------------------|

---

**Description**

Selects the row of a governed normative reference's model table that matches the supplied covariates: categorical stratification variables (e.g. sex) are matched exactly, numeric ones (e.g. age, speed) by nearest value. A covariate falling outside the tabulated range is flagged as an extrapolation.

**Usage**

```
matchStratum(ref, covariates)
```

**Arguments**

|            |                                                                      |
|------------|----------------------------------------------------------------------|
| ref        | A <a href="#">GovernedNormativeReference</a> .                       |
| covariates | Named list of covariate values covering the reference's strata_vars. |

**Value**

A one-row `data.frame` (the matched stratum) with an "extrapolation" attribute (logical).

**See Also**

`normativeZScore()`, `normativeDeviation()`

**Examples**

```
ref <- GovernedNormativeReference("gs", "gait", "gait_speed",
  provenance = list(source = "x"), consent = list(status = "public"),
  license = list(spdx = "CC0-1.0"),
  governance = list(custodian = "lab", access_level = "open"),
  strata_vars = c("age", "sex"),
  model = list(type = "strata", table = data.frame(
    age = c(60, 70), sex = "M", mean = c(1.4, 1.3), sd = 0.2)))
matchStratum(ref, list(age = 68, sex = "M"))
```

---

mdcResponder

---

*Classify change scores against the minimal detectable change (MDC)*


---

**Description**

A first responder-analysis primitive: a change is a real improvement/decline only if it exceeds the MDC (measurement noise). Uses `PhysioCore::mdc`.

**Usage**

```
mdcResponder(change, sem_value, confidence = 0.95)
```

**Arguments**

|                         |                                                                        |
|-------------------------|------------------------------------------------------------------------|
| <code>change</code>     | Numeric change score(s) (follow-up minus baseline).                    |
| <code>sem_value</code>  | The standard error of measurement (see <code>PhysioCore::sem</code> ). |
| <code>confidence</code> | Confidence level for the MDC (default 0.95).                           |

**Value**

Character vector: "improved", "stable", or "declined".

**Examples**

```
mdcResponder(c(5, 0.5, -6), sem_value = 1.5)
```

---

|                    |                                                      |
|--------------------|------------------------------------------------------|
| normativeDeviation | <i>Batch normative deviation over a metric table</i> |
|--------------------|------------------------------------------------------|

---

### Description

Applies [normativeZScore](#) to every row of a metric table: each row supplies a value plus the covariate columns named by the reference's strata\_vars.

### Usage

```
normativeDeviation(metric_table, ref, deviation_z = 2)
```

### Arguments

|              |                                                                                  |
|--------------|----------------------------------------------------------------------------------|
| metric_table | A data.frame with a value column and one column per ref stratification variable. |
| ref          | A <a href="#">GovernedNormativeReference</a> .                                   |
| deviation_z  | Deviation-flag threshold (see <a href="#">normativeZScore()</a> ).               |

### Value

metric\_table with added z, percentile, deviation\_flag and extrapolation columns.

### See Also

[normativeZScore\(\)](#)

---

|                 |                                                          |
|-----------------|----------------------------------------------------------|
| normativeZScore | <i>Normative z-score / percentile for an observation</i> |
|-----------------|----------------------------------------------------------|

---

### Description

Standardizes an observed value against a governed normative reference, selecting the covariate-matched stratum and applying either a Gaussian  $(value - \mu)/\sigma$  (stratified mean/sd model) or the Cole (1990) LMS transform (LMS model). An unmatched or unsupported stratum raises an error rather than returning a silent NA.

### Usage

```
normativeZScore(value, ref, covariates = list(), deviation_z = 2)
```

**Arguments**

|             |                                                                                                                             |
|-------------|-----------------------------------------------------------------------------------------------------------------------------|
| value       | Numeric observed value.                                                                                                     |
| ref         | A <a href="#">GovernedNormativeReference</a> .                                                                              |
| covariates  | Named list of covariates (e.g. <code>list(age = 68, sex = "M")</code> ) covering the reference's <code>strata_vars</code> . |
| deviation_z | Absolute z above which <code>deviation_flag</code> is set (default 2).                                                      |

**Value**

A list with `z`, percentile (0-100), `deviation_flag` and extrapolation.

**References**

Cole TJ (1990). The LMS method for constructing normalized growth standards. *Eur J Clin Nutr* 44(1), 45-60.

**See Also**

[matchStratum\(\)](#), [normativeDeviation\(\)](#)

**Examples**

```
ref <- GovernedNormativeReference("gs", "gait", "gait_speed",
  provenance = list(source = "x"), consent = list(status = "public"),
  license = list(spdx = "CC0-1.0"),
  governance = list(custodian = "lab", access_level = "open"),
  strata_vars = "sex",
  model = list(type = "strata",
    table = data.frame(sex = "M", mean = 1.3, sd = 0.2)))
normativeZScore(1.0, ref, list(sex = "M"))
```

---

|                           |                                                                      |
|---------------------------|----------------------------------------------------------------------|
| <code>promisRawToT</code> | <i>Convert a PROMIS summed score to a T-score via a lookup table</i> |
|---------------------------|----------------------------------------------------------------------|

---

**Description**

The other official PROMIS scoring path for a full short form: map the summed raw score to a T-score with the instrument's published summed-score-to-T-score table. The table is not bundled - obtain it from the HealthMeasures scoring manual for the specific short form.

**Usage**

```
promisRawToT(raw, conversion_table)
```

**Arguments**

`raw` Summed raw score.  
`conversion_table` A data.frame with a `raw` column, a `tscore` column and optionally an `se` column (the published conversion table).

**Value**

A list with `tscore` and `se` (NA if the table has no `se` column); NA with a warning if `raw` is not in the table.

**See Also**

[scorePROMIS\(\)](#)

**Examples**

```
tab <- data.frame(raw = 4:8, tscore = c(21.5, 30.1, 35.7, 40.2, 44.0))
promisRawToT(6, tab)
```

---

raschAnalyze

*Rasch (interval) analysis of a clinical ADL/IADL instrument*

---

**Description**

Fits a polytomous Rasch model to a cohort's responses on a registered instrument and returns the psychometric summary that published ADL/IADL analyses report: the item difficulty hierarchy, category threshold ordering (with disordered-threshold flags), person and item measures on an interval (logit) scale, item fit (infit/outfit), person/item separation reliability, and the raw-score to interval-measure conversion. The Partial Credit Model is the default (it accommodates the Barthel Index's mixed item lengths); pass `model = "RSM"` for a uniform-format scale.

**Usage**

```
raschAnalyze(instrument, responses, model = c("PCM", "RSM"), ...)
```

**Arguments**

`instrument` A [ClinicalInstrument](#) or an instrument id (e.g. "barthel", "lawton\_iadl").  
`responses` A persons x items matrix or data.frame of responses on the instrument's native scale (see [raschRecode\(\)](#)).  
`model` "PCM" (default) or "RSM".  
`...` Passed to [PhysioAppKit::pcm\\_measure\(\)](#).

**Details**

Requires the suggested package **PhysioAppKit** (the domain-neutral engine).

Value

The poly\_rasch fit (see `PhysioAppKit::pcm_measure()`) with an added class `clin_rasch` and extra fields: `instrument` (id), `item_hierarchy` (items ordered hardest-to-easiest by calibrated location) and `recoded` (the category matrix analysed).

See Also

`raschRecode()`, `scoreBarthel()`, `PhysioAppKit::pcm_measure()`

Examples

```
set.seed(1)
items <- getInstrument("barthel")@items
# a small synthetic cohort of weighted Barthel responses
resp <- t(replicate(50, {
  ability <- stats::rnorm(1)
  vapply(c(2, 1, 1, 2, 2, 2, 2, 3, 3, 2), function(m)
    c(0, 5, 10, 15)[min(m, max(0, round(ability + m / 2))) + 1], numeric(1))
}))
colnames(resp) <- items
if (requireNamespace("PhysioAppKit", quietly = TRUE)) {
  fit <- raschAnalyze("barthel", resp)
  fit$item_hierarchy
}
```

---

|             |                                                                             |
|-------------|-----------------------------------------------------------------------------|
| raschRecode | <i>Recode clinical instrument responses to consecutive Rasch categories</i> |
|-------------|-----------------------------------------------------------------------------|

---

Description

Turns a persons x items table of responses on an instrument’s native scale (e.g. the weighted Barthel levels 0/5/10/15) into the consecutive integer categories 0..m that a Rasch model expects, using the instrument’s declared admissible `item_values` as the category order.

Usage

```
raschRecode(instrument, responses)
```

Arguments

- `instrument` A [ClinicalInstrument](#) or an instrument id.
- `responses` A persons x items matrix or data.frame; columns are matched to the instrument’s items by name, or taken in the instrument’s item order if unnamed.

Value

An integer matrix of consecutive category codes (0..m per item), with the instrument’s item names as columns.

## See Also

[raschAnalyze\(\)](#)

## Examples

```
resp <- rbind(c(10, 5, 5, 10, 10, 10, 10, 15, 15, 10),
              c(5, 0, 0, 5, 5, 5, 5, 10, 5, 0))
colnames(resp) <- getInstrument("barthel")@items
raschRecode("barthel", resp)
```

---

raschResponder

*Responder analysis of pre-to-post change on the Rasch interval scale*

---

## Description

Classifies each person's change between two occasions using interval (logit) measures from [raschAnalyze\(\)](#) / [PhysioAppKit::pcm\\_measure\(\)](#). The Rasch model's own standard errors give a distribution-free standard error of measurement and hence the Minimal Detectable Change; the pre/post change is then run through [mdcResponder\(\)](#) (reliable change) and, when an MCID on the logit scale is supplied, [classifyResponder\(\)](#) (the MDC x MCID framework).

## Usage

```
raschResponder(
  pre,
  post,
  mcid = NULL,
  confidence = 0.95,
  direction = c("increase", "decrease")
)
```

## Arguments

|            |                                                                                                                                                              |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| pre, post  | Baseline and follow-up measures: a poly_rasch fit, a list with theta/theta_se, or a numeric measure vector. se is required (from the fit) to derive the MDC. |
| mcid       | Optional MCID on the logit scale; enables the four-level <a href="#">classifyResponder()</a> output.                                                         |
| confidence | Confidence level for the MDC (default 0.95).                                                                                                                 |
| direction  | "increase" (default; higher measure = better) or "decrease".                                                                                                 |

## Details

The two occasions must be on a common metric – calibrate the items once and anchor them, or co-calibrate – and the persons must align by row.

**Value**

a data.frame, one row per person: pre, post, change, improvement (direction-aware), sem, mdc, reliable\_change (from [mdcResponder\(\)](#)) and, if mcid given, classification (from [classifyResponder\(\)](#)). Attributes sem, mdc, mcid record the values used.

**See Also**

[raschAnalyze\(\)](#), [mdcResponder\(\)](#), [classifyResponder\(\)](#), [estimateMDC\(\)](#)

**Examples**

```
pre <- list(theta = c(0, 0, 0, 0), theta_se = rep(0.3, 4))
post <- list(theta = c(1.5, 0.1, -1.2, 0.05), theta_se = rep(0.3, 4))
raschResponder(pre, post, mcid = 1.0)
```

---

|                    |                                       |
|--------------------|---------------------------------------|
| registerInstrument | <i>Register a clinical instrument</i> |
|--------------------|---------------------------------------|

---

**Description**

Adds an instrument to the registry, either a [ClinicalInstrument](#) object, a parsed-spec list, or the path to a YAML spec file.

**Usage**

```
registerInstrument(spec, overwrite = FALSE)
```

**Arguments**

|           |                                                                                              |
|-----------|----------------------------------------------------------------------------------------------|
| spec      | A <a href="#">ClinicalInstrument</a> , a spec list, or a path to a .yaml file.               |
| overwrite | Replace an existing instrument with the same id? Default FALSE (an existing id is an error). |

**Value**

Invisibly, the registered [ClinicalInstrument](#).

**See Also**

[getInstrument\(\)](#), [listInstruments\(\)](#)

**Examples**

```
inst <- ClinicalInstrument(id = "toy2", items = c("a", "b"),
  item_ranges = list(a = c(0, 1), b = c(0, 1)))
registerInstrument(inst)
```

---

|                   |                                                                 |
|-------------------|-----------------------------------------------------------------|
| registerNormative | <i>Register a normative reference in the versioned registry</i> |
|-------------------|-----------------------------------------------------------------|

---

## Description

Persists a validated [GovernedNormativeReference](#) to the registry at `<root>/<id>/<version>.rds`, writing a sibling governance manifest.json. The artifact must pass both object validity and [validateNormativeManifest](#).

## Usage

```
registerNormative(ref, root = NULL, overwrite = FALSE)
```

## Arguments

|           |                                                                                                                            |
|-----------|----------------------------------------------------------------------------------------------------------------------------|
| ref       | A <a href="#">GovernedNormativeReference</a> .                                                                             |
| root      | Registry root directory; defaults to the option <code>PhysioClinical.normative_root</code> or the package's user data dir. |
| overwrite | Overwrite an already-registered version (default FALSE).                                                                   |

## Value

The written .rds path, invisibly.

## See Also

[getNormative\(\)](#), [listNormative\(\)](#)

## Examples

```
ref <- GovernedNormativeReference("gs", "gait", "gait_speed",
  provenance = list(source = "x"), consent = list(status = "public"),
  license = list(spdx = "CC0-1.0"),
  governance = list(custodian = "lab", access_level = "open"))
registerNormative(ref, root = tempfile("norm"))
```

---

|             |                                                                           |
|-------------|---------------------------------------------------------------------------|
| romNormalcy | <i>Compare a measured joint ROM to the reference (and the other side)</i> |
|-------------|---------------------------------------------------------------------------|

---

## Description

Compares a measured joint range of motion against the packaged reference angle (see [romReference\(\)](#)) and, when supplied, the contralateral (usually unaffected) side - the two comparisons clinicians read a goniometry measurement against.

Usage

```
romNormalcy(measured, joint, motion, contralateral = NULL)
```

Arguments

- measured            Measured range of motion, in degrees.
- joint, motion      The joint and motion (case insensitive); must identify a single reference row.
- contralateral      Optional measured ROM of the other side, in degrees.

Value

A one-row data.frame (class "rom\_normalcy") with reference\_deg, percent\_of\_normal (NA for an extension-to-neutral motion whose reference is 0 deg), deficit\_vs\_reference (reference minus measured) and limited (measured below reference); plus contralateral\_deg, deficit\_vs\_contralateral and percent\_of\_contralateral when a contralateral value is given.

See Also

```
romReference()
```

Examples

```
romNormalcy(100, "knee", "flexion")
romNormalcy(100, "knee", "flexion", contralateral = 130)
```

---

|              |                                                            |
|--------------|------------------------------------------------------------|
| romReference | <i>Joint range-of-motion reference values (goniometry)</i> |
|--------------|------------------------------------------------------------|

---

Description

Returns the packaged normal active range-of-motion reference values for the major peripheral joints (shoulder, elbow, forearm, wrist, hip, knee, ankle). These are the consensus clinical goniometry reference angles (in degrees) of the American Academy of Orthopaedic Surgeons, as tabulated by Norkin & White, *Measurement of Joint Motion*. They are single reference angles (not a distribution), so use `romNormalcy()` for a reference / side-to-side comparison; for a stratified z-score build a `GovernedNormativeReference()` from a cohort dataset and use `normativeZScore()`.

Usage

```
romReference(joint = NULL, motion = NULL)
```

Arguments

- joint                Optional joint to filter by (e.g. "knee"; case insensitive).
- motion              Optional motion to filter by (e.g. "flexion"; case insensitive).

Value

A data.frame with joint, motion, plane, reference\_deg and source.

References

American Academy of Orthopaedic Surgeons (1965). *Joint Motion: Method of Measuring and Recording*. Norkin CC, White DJ. *Measurement of Joint Motion: A Guide to Goniometry*. F.A. Davis.

See Also

[romNormalcy\(\)](#), [normativeZScore\(\)](#)

Examples

```
romReference("knee")
romReference(motion = "flexion")
```

---

|            |                                           |
|------------|-------------------------------------------|
| score10MWT | <i>Score a 10-metre walk test (10mWT)</i> |
|------------|-------------------------------------------|

---

Description

Computes comfortable/fast gait speed and its ambulation category, with an optional normative z-score. Accepts a gait speed directly, a walk time, or a PhysioMoCap signal-derived result (walk\_test\_report from PhysioMoCap::instrumented10mWT or gait\_parameters from PhysioMoCap::calculateGaitParameters).

Usage

```
score10MWT(
  x = NULL,
  time_s = NULL,
  distance = 10,
  pace = c("comfortable", "fast"),
  ref = NULL,
  covariates = list()
)
```

Arguments

|            |                                                                                      |
|------------|--------------------------------------------------------------------------------------|
| x          | A gait speed (m/s), a walk_test_report, or a gait_parameters; or NULL to use time_s. |
| time_s     | Walk time in seconds (manual path; speed = distance/time).                           |
| distance   | Walk distance in metres (default 10).                                                |
| pace       | "comfortable" (default) or "fast".                                                   |
| ref        | Optional <a href="#">GovernedNormativeReference</a> for the normative z-score.       |
| covariates | Covariates (e.g. list(age, sex)) for ref.                                            |

**Value**

A "performance\_test\_score" with gait\_speed, ambulation and (if ref) zscore.

**References**

Perry et al. (1995); Fritz & Lusardi (2009); Perera et al. (2006).

**See Also**

[score6MWT\(\)](#), [scoreTUG\(\)](#)

**Examples**

```
score10MWT(time_s = 8)          # 10 m / 8 s = 1.25 m/s -> community ambulator
```

---

|           |                                          |
|-----------|------------------------------------------|
| score6MWT | <i>Score a 6-minute walk test (6MWT)</i> |
|-----------|------------------------------------------|

---

**Description**

Reports the walked distance with, when the demographics are supplied, the Enright & Sherrill (1998) predicted distance, percent-predicted and the lower limit of normal (LLN), plus an optional normative z-score.

**Usage**

```
score6MWT(
  distance_m,
  age = NULL,
  sex = NULL,
  height_cm = NULL,
  weight_kg = NULL,
  ref = NULL,
  covariates = list()
)
```

**Arguments**

distance\_m      Distance walked in 6 minutes (metres).  
age, sex, height\_cm, weight\_kg      Demographics for the Enright reference equation (all four required for the predicted value).  
ref, covariates      Optional normative reference / covariates for a z-score.

**Value**

A "performance\_test\_score" with distance, and (when demographics are given) predicted, percent\_predicted, lln and below\_lln.

## References

Enright PL, Sherrill DL (1998). *Am J Respir Crit Care Med* 158(5), 1384-1387. doi:[10.1164/ajrccm.158.5.9710086](https://doi.org/10.1164/ajrccm.158.5.9710086)

## See Also

[score10MWT\(\)](#), [scoreTUG\(\)](#)

## Examples

```
score6MWT(450, age = 60, sex = "male", height_cm = 175, weight_kg = 75)
```

---

scoreARAT

*Score the Action Research Arm Test (ARAT)*

---

## Description

Score the Action Research Arm Test (ARAT)

## Usage

```
scoreARAT(items, ...)
```

## Arguments

|       |                                                               |
|-------|---------------------------------------------------------------|
| items | Named numeric responses (0-3 per item) for the 19 ARAT items. |
| ...   | Passed to <a href="#">scoreInstrument()</a> .                 |

## Value

A [ClinicalScore](#) with the total (0-57) and the grasp/grip/pinch/ gross-movement subscale scores.

## See Also

[scoreFMAUE\(\)](#)

---

|              |                                                                   |
|--------------|-------------------------------------------------------------------|
| scoreBarthel | <i>Score the Barthel Index (basic activities of daily living)</i> |
|--------------|-------------------------------------------------------------------|

---

### Description

The Barthel Index rates ten basic ADL domains on their conventional weighted scale (steps of 5; transfers and mobility to 15) for a total of 0-100, higher being more independent.

### Usage

```
scoreBarthel(items, ...)
```

### Arguments

|       |                                                                                                                                                                                                                                                               |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| items | Named numeric responses on the weighted per-item scale (e.g. feeding = 10, bathing = 5, transfers = 15), or an unnamed vector in the instrument's item order. Allowed values are the conventional weights (0/5/10, and 0/5/10/15 for transfers and mobility). |
| ...   | Passed to <a href="#">scoreInstrument()</a> (e.g. missing, subject_id).                                                                                                                                                                                       |

### Value

A [ClinicalScore](#) with the total (0-100) and the dependence stratum (total / severe / moderate / slight dependence / independent).

### See Also

[scoreKatz\(\)](#), [scoreLawton\(\)](#), [raschAnalyze\(\)](#)

### Examples

```
scoreBarthel(stats::setNames(
  c(10, 5, 5, 10, 10, 10, 10, 15, 15, 10), getInstrument("barthel")@items))
```

---

|           |                                           |
|-----------|-------------------------------------------|
| scoreBerg | <i>Score the Berg Balance Scale (BBS)</i> |
|-----------|-------------------------------------------|

---

### Description

Score the Berg Balance Scale (BBS)

### Usage

```
scoreBerg(items, ...)
```

**Arguments**

items            Named numeric responses (0-4 per item) for the 14 Berg items.  
 ...              Passed to `scoreInstrument()`.

**Value**

A `ClinicalScore` with the total (0-56) and the fall-risk stratum.

**See Also**

`scoreFMALE()`

---

|          |                                                    |
|----------|----------------------------------------------------|
| scoreBPI | <i>Score the Brief Pain Inventory (short form)</i> |
|----------|----------------------------------------------------|

---

**Description**

Scores the BPI severity (worst / least / average / current pain) and interference (general activity, mood, walking, work, relations, sleep, enjoyment of life) items, each 0-10. Severity and interference are reported as their mean subscale scores (each 0-10, the conventional BPI summaries); the total is the grand mean over all items.

**Usage**

```
scoreBPI(items, ...)
```

**Arguments**

items            Named numeric responses (e.g. worst\_pain = 8, interference\_sleep = 5) or an unnamed vector in the instrument's item order.  
 ...              Passed to `scoreInstrument()` (e.g. missing = "prorate").

**Value**

A `ClinicalScore` whose subscales hold the severity and interference means.

**See Also**

`scorePainNRS()`, `getInstrument()`

**Examples**

```
scoreBPI(stats::setNames(c(8, 2, 5, 4, 5, 6, 5, 7, 3, 5, 6),
  getInstrument("bpi")@items))
```

---

|             |                                                                                  |
|-------------|----------------------------------------------------------------------------------|
| scoreCohort | <i>Score a cohort's instrument responses into interoper-ready ClinicalScores</i> |
|-------------|----------------------------------------------------------------------------------|

---

## Description

Scores a per-subject table of instrument responses into a list of [ClinicalScore](#) objects, each tagged with its `subject_id`, ready to drive the FHIR / OMOP / CDISC exporters ([writeFHIRBundle\(\)](#), [toOMOP\(\)](#), [toCDISC\\_QS\(\)](#)) directly. The `subject_col` and `item` columns can come straight from a cohort's data (e.g. a `PhysioCohort` subject table joined to captured responses).

## Usage

```
scoreCohort(
  responses,
  instrument,
  subject_col = "subject_id",
  timestamp_col = NULL,
  missing = c("error", "prorate", "na")
)
```

## Arguments

|                            |                                                                                                                                                                                   |
|----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>responses</code>     | A <code>data.frame</code> : one row per subject, with a subject-id column and one column per instrument item (columns not matching an item are ignored).                          |
| <code>instrument</code>    | A <a href="#">ClinicalInstrument</a> or an instrument id (resolved via <a href="#">getInstrument()</a> ).                                                                         |
| <code>subject_col</code>   | Name of the subject-id column (default <code>"subject_id"</code> ).                                                                                                               |
| <code>timestamp_col</code> | Optional name of a timestamp column stored on each score.                                                                                                                         |
| <code>missing</code>       | Missing-data policy passed to <a href="#">scoreInstrument()</a> (default <code>"error"</code> ; use <code>"prorate"</code> / <code>"na"</code> when only some items are present). |

## Value

A named list (by subject id) of [ClinicalScore](#) objects.

## See Also

[scoreInstrument\(\)](#), [writeFHIRBundle\(\)](#), [toOMOP\(\)](#), [toCDISC\\_QS\(\)](#)

## Examples

```
items <- getInstrument("katz_adl")@items
resp <- data.frame(subject_id = c("P01", "P02"),
  stats::setNames(as.data.frame(rbind(rep(1, 6), c(1, 0, 1, 1, 0, 1))), items))
scores <- scoreCohort(resp, "katz_adl")
vapply(scores, function(s) s$total, numeric(1))
```

---

|             |                                                                |
|-------------|----------------------------------------------------------------|
| scoreEQ5D5L | <i>Score the EQ-5D-5L descriptive system (level-sum score)</i> |
|-------------|----------------------------------------------------------------|

---

**Description**

Sums the five EQ-5D-5L dimension levels (1 = no problems to 5 = extreme problems / unable) to a level-sum score of 5-25, higher indicating worse health. The utility index requires a country-specific value set and is not computed here; the EQ-VAS (0-100) is a separate rating.

**Usage**

scoreEQ5D5L(items, ...)

**Arguments**

- items      Named numeric levels (e.g. mobility = 2, pain\_discomfort = 3) or an unnamed vector in the instrument's item order (mobility, self-care, usual activities, pain/discomfort, anxiety/depression).
- ...      Passed to `scoreInstrument()`.

**Value**

A `ClinicalScore` with the level-sum score (5-25).

**See Also**

`getInstrument()`

**Examples**

```
scoreEQ5D5L(c(mobility = 2, self_care = 1, usual_activities = 2,
              pain_discomfort = 3, anxiety_depression = 2))
```

---

|          |                                                          |
|----------|----------------------------------------------------------|
| scoreFAM | <i>Score the Functional Assessment Measure (FIM+FAM)</i> |
|----------|----------------------------------------------------------|

---

**Description**

Score the Functional Assessment Measure (FIM+FAM)

**Usage**

scoreFAM(items, ...)

Arguments

- items            Named numeric responses (1-7 per item) for the 30 FIM+FAM items.
- ...             Passed to `scoreInstrument()`.

Value

A `ClinicalScore` with the total (30-210) and the combined motor (16 items) and cognitive (14 items) subscale scores.

See Also

`scoreFIM()`

---

|          |                                                        |
|----------|--------------------------------------------------------|
| scoreFIM | <i>Score the Functional Independence Measure (FIM)</i> |
|----------|--------------------------------------------------------|

---

Description

Score the Functional Independence Measure (FIM)

Usage

```
scoreFIM(items, ...)
```

Arguments

- items            Named numeric responses (1-7 per item) for the 18 FIM items.
- ...             Passed to `scoreInstrument()`.

Value

A `ClinicalScore` with the total (18-126) and subscale scores for the six FIM domains (self-care, sphincter, transfers, locomotion, communication, social cognition) plus the combined motor (13 items) and cognitive (5 items) scores.

See Also

`scoreFAM()`

Examples

```
scoreFIM(stats::setNames(rep(4, 18), getInstrument("fim")@items))
```

scoreFMALE

*Score the Fugl-Meyer Assessment, Lower Extremity (FMA-LE)***Description**

Score the Fugl-Meyer Assessment, Lower Extremity (FMA-LE)

**Usage**

```
scoreFMALE(items, ...)
```

**Arguments**

|       |                                                                                                                            |
|-------|----------------------------------------------------------------------------------------------------------------------------|
| items | Named numeric responses (0-2 per item) for the 17 FMA-LE motor items, or an unnamed vector in the instrument's item order. |
| ...   | Passed to <a href="#">scoreInstrument()</a> .                                                                              |

**Value**

A [ClinicalScore](#) with the total (0-34) and the motor and coordination/speed subscale scores.

**See Also**

[scoreBerg\(\)](#), [scoreFIM\(\)](#)

**Examples**

```
scoreFMALE(stats::setNames(rep(2, 17), getInstrument("fma_le")@items))
```

scoreFMAUE

*Score the Fugl-Meyer Assessment, Upper Extremity (FMA-UE)***Description**

Score the Fugl-Meyer Assessment, Upper Extremity (FMA-UE)

**Usage**

```
scoreFMAUE(items, ...)
```

**Arguments**

|       |                                                                                                                            |
|-------|----------------------------------------------------------------------------------------------------------------------------|
| items | Named numeric responses (0-2 per item) for the 33 FMA-UE motor items, or an unnamed vector in the instrument's item order. |
| ...   | Passed to <a href="#">scoreInstrument()</a> (e.g. missing, subject_id).                                                    |



scoreGAS

*Score Goal Attainment Scaling (GAS T-score)***Description**

Computes the standardized GAS T-score across a set of goals given their attained levels:

$$T = 50 + \frac{10 \sum_i w_i x_i}{\sqrt{(1 - \rho) \sum_i w_i^2 + \rho (\sum_i w_i)^2}}$$

where  $x_i$  is the attained level,  $w_i$  the weight and  $\rho$  the assumed inter-goal correlation. All goals at their expected level ( $x_i = 0$ ) give  $T = 50$ .

**Usage**

```
scoreGAS(goals, attained_levels, rho = 0.3)
```

**Arguments**

`goals` A `defineGoal` object or a list of them.

`attained_levels` Numeric attained level for each goal (each within that goal's levels).

`rho` Assumed inter-goal correlation in  $[0, 1)$ ; default 0.3.

**Value**

A "gas\_result" with the T-score, per-goal weights/levels and rho.

**References**

Kiresuk & Sherman (1968); Turner-Stokes (2009).

**See Also**

`defineGoal()`, `gasSummary()`

**Examples**

```
g1 <- defineGoal("Transfers", importance = 3, difficulty = 2)
g2 <- defineGoal("Stairs", importance = 2, difficulty = 3)
scoreGAS(list(g1, g2), attained_levels = c(1, 0))
```

---

|                 |                                                           |
|-----------------|-----------------------------------------------------------|
| scoreInstrument | <i>Score item responses against a clinical instrument</i> |
|-----------------|-----------------------------------------------------------|

---

## Description

Validates a set of item responses against a [ClinicalInstrument](#), aggregates them into the total and subscale scores under the chosen missing-data policy, and assigns the interpretation stratum.

## Usage

```
scoreInstrument(
  instrument,
  items,
  missing = c("error", "prorate", "na"),
  subject_id = NA_character_,
  timestamp = NA_character_
)
```

## Arguments

|            |                                                                                                                                                                                        |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| instrument | A <a href="#">ClinicalInstrument</a> , or an instrument id resolved via <a href="#">getInstrument()</a> .                                                                              |
| items      | A named numeric vector (item id -> response), a one-row data.frame, or an unnamed vector in the instrument's item order.                                                               |
| missing    | Missing-data policy: "error" (default; any missing item is an error), "prorate" (scale the observed items up to the full count) or "na" (return NA for any score with a missing item). |
| subject_id | Optional subject identifier stored on the result.                                                                                                                                      |
| timestamp  | Optional timestamp stored on the result.                                                                                                                                               |

## Value

A [ClinicalScore](#).

## See Also

[ClinicalInstrument](#), [getInstrument\(\)](#)

## Examples

```
inst <- ClinicalInstrument(
  id = "toy", items = c("q1", "q2", "q3"),
  item_ranges = list(q1 = c(0, 4), q2 = c(0, 4), q3 = c(0, 4)))
scoreInstrument(inst, c(q1 = 2, q2 = 3, q3 = 4))
```

scoreKatz

*Score the Katz Index of Independence in ADL***Description**

Six basic ADL activities (bathing, dressing, toileting, transferring, continence, feeding), each scored independent (1) or dependent (0), for a total of 0-6.

**Usage**

```
scoreKatz(items, ...)
```

**Arguments**

|                    |                                                                                                 |
|--------------------|-------------------------------------------------------------------------------------------------|
| <code>items</code> | Named numeric responses (0/1 per activity) or an unnamed vector in the instrument's item order. |
| <code>...</code>   | Passed to <code>scoreInstrument()</code> .                                                      |

**Value**

A `ClinicalScore` with the total (0-6) and the impairment stratum.

**See Also**

`scoreBarthel()`, `scoreLawton()`

**Examples**

```
scoreKatz(c(bathing = 1, dressing = 1, toileting = 1,
            transferring = 1, continence = 0, feeding = 1))
```

scoreLawton

*Score the Lawton-Brody IADL scale (instrumental ADL)***Description**

Eight instrumental-ADL domains of independent community living (telephone, shopping, food preparation, housekeeping, laundry, transportation, medication, finances), each dichotomised independent (1) / dependent (0), summed to a 0-8 count. Lawton & Brody define no total-score severity bands, so the returned stratum is NA.

**Usage**

```
scoreLawton(items, ...)
```

**Arguments**

`items`            Named numeric responses (0/1 per domain) or an unnamed vector in the instrument's item order. For the abbreviated male version, omit the food-preparation, housekeeping and laundry items and use `missing = "na"`.

`...`            Passed to `scoreInstrument()`.

**Value**

A `ClinicalScore` with the total (0-8).

**See Also**

`scoreBarthel()`, `scoreKatz()`, `raschAnalyze()`

**Examples**

```
scoreLawton(stats::setNames(rep(1, 8), getInstrument("lawton_iadl")@items))
```

---

scoreMAS

*Score a Modified Ashworth Scale (MAS) rating*

---

**Description**

Accepts a MAS rating as its ordinal level label ("`0`", "`1`", "`1+`", "`2`", "`3`", "`4`") or the equivalent number, mapping "`1+`" to 1.5 for arithmetic while preserving the label. Rate one muscle group per call.

**Usage**

```
scoreMAS(rating, ...)
```

**Arguments**

`rating`            A single MAS level label or number.

`...`            Passed to `scoreInstrument()`.

**Value**

A `ClinicalScore` whose total is the numeric MAS value; the ordinal label is recovered with `masLevelLabel()`.

**See Also**

`masLevelLabel()`

**Examples**

```
scoreMAS("1+")
```

---

|           |                                                       |
|-----------|-------------------------------------------------------|
| scoreMFIS | <i>Score the Modified Fatigue Impact Scale (MFIS)</i> |
|-----------|-------------------------------------------------------|

---

**Description**

Sums the twenty-one MFIS items (each 0-4) to a total of 0-84 with physical (9-item), cognitive (10-item) and psychosocial (2-item) subscales. A total of 38 or more indicates fatigue.

**Usage**

```
scoreMFIS(items, ...)
```

**Arguments**

|       |                                                                                                 |
|-------|-------------------------------------------------------------------------------------------------|
| items | Named numeric responses (item01 .. item21) or an unnamed vector in the instrument's item order. |
| ...   | Passed to <a href="#">scoreInstrument()</a> (e.g. missing = "prorate").                         |

**Value**

A [ClinicalScore](#) with the total (0-84), the three subscale scores and the fatigue stratum.

**See Also**

[scoreFSS\(\)](#), [getInstrument\(\)](#)

**Examples**

```
scoreMFIS(stats::setNames(rep(2, 21), getInstrument("mfis")@items))
```

---

|           |                                                       |
|-----------|-------------------------------------------------------|
| scoreMMSE | <i>Score the Mini-Mental State Examination (MMSE)</i> |
|-----------|-------------------------------------------------------|

---

**Description**

Sums the eleven MMSE components (orientation to time and place, registration, attention/calculation, recall, naming, repetition, a three-stage command, reading, writing and figure copying) to a total of 0-30, higher being better cognition.

**Usage**

```
scoreMMSE(items, ...)
```

**Arguments**

|                    |                                                                                                                                                                                              |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>items</code> | Named numeric responses giving the points earned on each component (e.g. <code>orientation_time = 5</code> , <code>recall = 2</code> ), or an unnamed vector in the instrument's item order. |
| <code>...</code>   | Passed to <code>scoreInstrument()</code> (e.g. <code>missing</code> , <code>subject_id</code> ).                                                                                             |

**Value**

A **ClinicalScore** with the total (0-30), the six-domain subscale scores and the impairment stratum (no / mild / severe impairment).

**See Also**

`scoreMoCA()`, `getInstrument()`

**Examples**

```
scoreMMSE(stats::setNames(c(5, 5, 3, 5, 3, 2, 1, 3, 1, 1, 1),
                           getInstrument("mmse")@items))
```

---

scoreMoCA

*Score the Montreal Cognitive Assessment (MoCA)*

---

**Description**

Sums the seven MoCA domains (visuospatial/executive, naming, attention, language, abstraction, delayed recall and orientation) to a total of 0-30. A total below 26 indicates cognitive impairment. One point is conventionally added for participants with 12 or fewer years of formal education (total capped at 30); apply that adjustment to the orientation item (or the total) before or after scoring as appropriate for your protocol.

**Usage**

```
scoreMoCA(items, ...)
```

**Arguments**

|                    |                                                                                                                                                                                              |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>items</code> | Named numeric responses giving the points earned on each domain (e.g. <code>delayed_recall = 3</code> , <code>orientation = 6</code> ), or an unnamed vector in the instrument's item order. |
| <code>...</code>   | Passed to <code>scoreInstrument()</code> .                                                                                                                                                   |

**Value**

A **ClinicalScore** with the total (0-30) and the impairment stratum (normal / mild / moderate-severe impairment).

**See Also**

[scoreMMSE\(\)](#), [getInstrument\(\)](#)

**Examples**

```
scoreMoCA(stats::setNames(c(5, 3, 6, 3, 2, 5, 6),
                           getInstrument("moca")@items))
```

---

scoreMRCSum

*Score the Medical Research Council (MRC) sum score*

---

**Description**

Sums manual muscle-test grades (0 = no contraction to 5 = normal power) for six movements tested bilaterally - shoulder abduction, elbow flexion, wrist extension, hip flexion, knee extension and ankle dorsiflexion - to a total of 0-60, higher being stronger. A total below 48 defines ICU-acquired weakness.

**Usage**

```
scoreMRCSum(items, ...)
```

**Arguments**

|       |                                                                                                                                |
|-------|--------------------------------------------------------------------------------------------------------------------------------|
| items | Named numeric grades (e.g. elbow_flexion_r = 4, ankle_dorsiflexion_l = 3) or an unnamed vector in the instrument's item order. |
| ...   | Passed to <a href="#">scoreInstrument()</a> (e.g. missing = "prorate").                                                        |

**Value**

A [ClinicalScore](#) with the total (0-60), the upper-limb / lower-limb / left / right subscale sums and the weakness stratum.

**See Also**

[getInstrument\(\)](#)

**Examples**

```
scoreMRCSum(stats::setNames(rep(5, 12), getInstrument("mrc_sum")@items))
```

---

|          |                                              |
|----------|----------------------------------------------|
| scoreMRS | <i>Score the modified Rankin Scale (mRS)</i> |
|----------|----------------------------------------------|

---

**Description**

Score the modified Rankin Scale (mRS)

**Usage**

```
scoreMRS(grade, ...)
```

**Arguments**

|       |                                               |
|-------|-----------------------------------------------|
| grade | A single global disability grade, 0-6.        |
| ...   | Passed to <a href="#">scoreInstrument()</a> . |

**Value**

A [ClinicalScore](#) whose total is the grade and whose stratum is the disability label.

**See Also**

[scoreNIHSS\(\)](#)

**Examples**

```
scoreMRS(3)
```

---

|            |                                           |
|------------|-------------------------------------------|
| scoreNIHSS | <i>Score the NIH Stroke Scale (NIHSS)</i> |
|------------|-------------------------------------------|

---

**Description**

Score the NIH Stroke Scale (NIHSS)

**Usage**

```
scoreNIHSS(items, ...)
```

**Arguments**

|       |                                                                                  |
|-------|----------------------------------------------------------------------------------|
| items | Named numeric responses for the 15 NIHSS components (each within its own range). |
| ...   | Passed to <a href="#">scoreInstrument()</a> .                                    |

**Value**

A [ClinicalScore](#) with the total (0-42) and the severity band (no stroke / minor / moderate / moderate-to-severe / severe).

**See Also**

[scoreMRS\(\)](#)

---

scorePainNRS

*Score a Pain Numeric Rating Scale (0-10)*

---

**Description**

A single 0-10 pain-intensity rating, banded into the conventional none / mild / moderate / severe strata. An equivalent 0-100 mm Visual Analogue Scale measures the same construct on a wider range.

**Usage**

```
scorePainNRS(items, ...)
```

**Arguments**

|                    |                                                                                             |
|--------------------|---------------------------------------------------------------------------------------------|
| <code>items</code> | The pain rating: a length-1 numeric, or a named vector <code>c(pain_intensity = x)</code> . |
| <code>...</code>   | Passed to <a href="#">scoreInstrument()</a> (e.g. <code>subject_id</code> ).                |

**Value**

A [ClinicalScore](#) with the rating (0-10) and its severity stratum.

**See Also**

[scoreBPI\(\)](#)

**Examples**

```
scorePainNRS(c(pain_intensity = 6))
```

scorePROMIS

*Score a PROMIS (or other GRM) short form by EAP***Description**

Estimates the latent trait for a set of item responses under Samejima's graded response model by expected a posteriori (EAP) scoring over a normal quadrature, and reports it on the PROMIS T-score metric (mean 50, SD 10).

**Usage**

```
scorePROMIS(
  responses,
  calibration,
  quad_points = 61,
  quad_range = 4,
  prior_mean = 0,
  prior_sd = 1
)
```

**Arguments**

|                         |                                                                                                                                                                                                           |
|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| responses               | Named numeric vector of item responses (category 1..K per item; NA for a skipped item), or a one-row data.frame. Names must match calibration\$item; an unnamed vector is taken in calibration row order. |
| calibration             | A data.frame with columns item, a (slope) and ordered threshold columns b1, b2, ... (a K-category item uses K-1 thresholds; pad shorter items with NA).                                                   |
| quad_points, quad_range | EAP quadrature: number of nodes and half-width (nodes span [-quad_range, quad_range]; defaults 61 and 4).                                                                                                 |
| prior_mean, prior_sd    | Normal prior on the latent trait (default standard normal, the PROMIS calibration metric).                                                                                                                |

**Details**

No item parameters are bundled: supply the official PROMIS calibration (slopes and thresholds) from HealthMeasures. Any GRM-scored instrument works with the same function.

**Value**

A list with theta, se\_theta, tscore ( $50 + 10 * \text{theta}$ ), se\_tscore and n\_items (items actually scored).

**References**

Samejima F (1969). Estimation of latent ability using a response pattern of graded scores. *Psychometrika Monograph* 17. PROMIS scoring: HealthMeasures, [www.healthmeasures.net](http://www.healthmeasures.net).

See Also

[promisRawToT\(\)](#)

Examples

```
# illustrative calibration (NOT official PROMIS parameters)
cal <- data.frame(item = c("i1", "i2", "i3"), a = c(2.4, 1.9, 2.7),
  b1 = c(-2, -1.5, -1.8), b2 = c(-1, -0.5, -0.7),
  b3 = c(0.2, 0.1, 0), b4 = c(1.4, 1.2, 1.1))
scorePROMIS(c(i1 = 4, i2 = 5, i3 = 4), cal)
```

---

|           |                                                |
|-----------|------------------------------------------------|
| scoreSF36 | <i>Score the SF-36 / RAND-36 Health Survey</i> |
|-----------|------------------------------------------------|

---

Description

Scores the RAND 36-Item Health Survey 1.0: each item’s raw response is recoded to 0-100 (via the instrument’s `item_recode` map) and averaged within its scale, giving the eight subscale scores (0-100, higher = better health) in the returned subscales: physical functioning, role-physical, role-emotional, energy/fatigue, emotional well-being, social functioning, pain and general health. The health-transition item is not part of the eight scales. The norm-based T-scores and the PCS/MCS component summaries use proprietary population weights and are out of scope.

Usage

```
scoreSF36(items, ...)
```

Arguments

- `items`                Named numeric raw responses on the questionnaire’s own answer scales (q01, q03..q36, omitting the health-transition item q02), or an unnamed vector in the instrument’s item order.
- `...`                Passed to [scoreInstrument\(\)](#) (e.g. `missing = "prorate"` to score a scale from its completed items).

Value

A [ClinicalScore](#) whose subscales hold the eight 0-100 scale scores (the total is the grand mean over all items, not a standard SF-36 score).

References

Hays RD, Sherbourne CD, Mazel RM (1993). The RAND 36-Item Health Survey 1.0. *Health Econ* 2:217-227.

See Also

[scoreEQ5D5L\(\)](#), [getInstrument\(\)](#)

Examples

```
# all best-health responses -> every subscale scores 100
best <- stats::setNames(
  c(1, rep(3, 10), rep(2, 7), 1, 1, 1, 1, 6, 6, 1, 1, 6, 6, 1, 6, 5, 5, 1, 5, 1),
  getInstrument("sf36")@items)
scoreSF36(best)@subscales
```

---

|          |                                           |
|----------|-------------------------------------------|
| scoreTUG | <i>Score a Timed Up and Go test (TUG)</i> |
|----------|-------------------------------------------|

---

Description

Classifies TUG time against the standard fall-risk cut-offs (< 10 s normal; >= 13.5 s elevated community-dwelling fall risk, Shumway-Cook 2000; >= 30 s likely dependent, Podsiadlo & Richardson 1991), with an optional normative z-score.

Usage

```
scoreTUG(x, ref = NULL, covariates = list())
```

Arguments

- x                    A TUG time in seconds, or a `PhysioMoCap i tug_report` (its `total_duration` is used).
- ref, covariates    Optional normative reference / covariates for a z-score.

Value

A "performance\_test\_score" with time, category and fall\_risk (logical, >= 13.5 s).

References

Podsiadlo & Richardson (1991); Shumway-Cook et al. (2000).

See Also

```
score10MWT\(\), score6MWT\(\)
```

Examples

```
scoreTUG(15)    # >= 13.5 s -> elevated fall risk
```

---

|           |                                                                               |
|-----------|-------------------------------------------------------------------------------|
| scoreWMFT | <i>Score the Wolf Motor Function Test functional-ability scale (WMFT-FAS)</i> |
|-----------|-------------------------------------------------------------------------------|

---

**Description**

Score the Wolf Motor Function Test functional-ability scale (WMFT-FAS)

**Usage**

```
scoreWMFT(fas, ...)
```

**Arguments**

|     |                                                                     |
|-----|---------------------------------------------------------------------|
| fas | Named numeric responses (0-5 per task) for the 15 functional tasks. |
| ... | Passed to <a href="#">scoreInstrument()</a> .                       |

**Value**

A [ClinicalScore](#) whose total is the mean functional-ability score.

**See Also**

[wmftMedianTime\(\)](#)

---

|                                        |                                                              |
|----------------------------------------|--------------------------------------------------------------|
| show,GovernedNormativeReference-method | <i>Show a normative reference (surfacing its governance)</i> |
|----------------------------------------|--------------------------------------------------------------|

---

**Description**

Show a normative reference (surfacing its governance)

**Usage**

```
## S4 method for signature 'GovernedNormativeReference'
show(object)
```

**Arguments**

|        |                                                |
|--------|------------------------------------------------|
| object | A <a href="#">GovernedNormativeReference</a> . |
|--------|------------------------------------------------|

**Value**

object, invisibly.

---

toADaM\_ADQS

Export clinical scores to an ADaM BDS (ADQS) data frame

---

### Description

Restructures a QS domain (or scores) into an ADaM Basic Data Structure analysis data set (one PARAMCD per test): AVAL/AVALC carry the numeric/character result and PARCAT1 the questionnaire category.

### Usage

```
toADaM_ADQS(x, ...)
```

### Arguments

**x** A [ClinicalScore](#), a list of them, or a QS-domain data frame from [toCDISC\\_QS\(\)](#).  
**...** Passed to [toCDISC\\_QS\(\)](#) when x is score(s).

### Value

An ADaM BDS data.frame (columns STUDYID, USUBJID, PARAMCD, PARAM, PARCAT1, AVAL, AVALC, AVISIT, AVISITN, ADT).

### References

CDISC ADaM Basic Data Structure (BDS).

### See Also

[toCDISC\\_QS\(\)](#)

---

toCDISC\_QS

Export clinical scores to a CDISC SDTM QS (Questionnaires) domain

---

### Description

Maps one or more [ClinicalScore](#) objects to a CDISC SDTMIG Questionnaires (QS) domain data frame: the total and each subscale becomes a QS record. QSTESTCD/QSTEST/QSCAT come from the packaged controlled-terminology map (cdisc\_ct.csv); an unmapped instrument gets a sponsor-defined QSTESTCD (upper-cased, forced to start with a letter, disambiguated against collisions, and truncated to the SDTM 8-character limit) with a warning. QSTESTCD values in the shipped map are sponsor-defined (ct\_status = "sponsor"), not asserted to be published CDISC CT.

**Usage**

```
toCDISC_QS(
  scores,
  ct_map = NULL,
  studyid = "STUDY",
  usubjid_map = NULL,
  visitnum = 1L,
  visit = NA_character_
)
```

**Arguments**

|                 |                                                                                                             |
|-----------------|-------------------------------------------------------------------------------------------------------------|
| scores          | A <a href="#">ClinicalScore</a> or a list of them.                                                          |
| ct_map          | Optional data frame overriding the packaged CT map (columns instrument, subscale, qstestcd, qstest, qscat). |
| studyid         | STUDYID value (default "STUDY").                                                                            |
| usubjid_map     | Optional named vector/list mapping subject_id to a USUBJID; otherwise USUBJID = <studyid>-<subject_id>.     |
| visitnum, visit | VISITNUM / VISIT for all records.                                                                           |

**Value**

A QS-domain data.frame conforming to the SDTMIG column layout.

**References**

CDISC SDTMIG QS domain.

**See Also**

[toADaM\\_ADQS\(\)](#), [toFHIRObservation\(\)](#), [toOMOP\(\)](#)

**Examples**

```
sc <- methods::new("ClinicalScore", instrument_id = "fim", total = 90,
  subscales = c(motor = 60, cognitive = 30),
  subject_id = "01-001", timestamp = "2026-07-26")
toCDISC_QS(sc)
```

---

|                   |                                                         |
|-------------------|---------------------------------------------------------|
| toFHIRObservation | <i>Export a clinical score as a FHIR R4 Observation</i> |
|-------------------|---------------------------------------------------------|

---

## Description

Serializes a [ClinicalScore](#) into an HL7 FHIR R4 Observation resource (as a **jsonlite**-ready list): a survey-category Observation whose code uses the LOINC map (falling back to a text-only code), the total as valueQuantity, and each subscale as a component. Provenance can be attached via derivedFrom.

## Usage

```
toFHIRObservation(
  score,
  subject = NULL,
  effectiveDateTime = NULL,
  derivedFrom = NULL
)
```

## Arguments

|                   |                                                                                                                                                                                                                           |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| score             | A <a href="#">ClinicalScore</a> .                                                                                                                                                                                         |
| subject           | Subject reference (e.g. "Patient/123"); used verbatim. Defaults to the score's subject_id: a bare id becomes "Patient/<id>", while an id that is already a reference (contains a "/" or a urn:/URL scheme) is used as-is. |
| effectiveDateTime | ISO-8601 datetime; defaults to the score's timestamp.                                                                                                                                                                     |
| derivedFrom       | Optional character vector of source references.                                                                                                                                                                           |

## Value

A "fhir\_observation" list.

## References

HL7 FHIR R4 Observation; LOINC.

## See Also

[writeFHIRBundle\(\)](#), [validateFHIRObservation\(\)](#)

## Examples

```
sc <- methods::new("ClinicalScore", instrument_id = "berg", total = 45,
  subject_id = "P01", timestamp = "2026-07-26T09:00:00Z")
toFHIRObservation(sc)
```

toOMOP

*Export clinical scores to OMOP CDM v5.4 tables***Description**

Maps one or more [ClinicalScore](#) objects to OMOP Common Data Model v5.4 MEASUREMENT and OBSERVATION rows. The total and each subscale become a row, routed to MEASUREMENT or OBSERVATION by the concept map's domain\_id. Concept ids are looked up in the packaged `omop_concept_map.csv`; an unmapped instrument yields `measurement_concept_id = 0` (the OMOP "No matching concept") with a warning. No Athena concept ids are fabricated — the shipped map leaves them blank; supply a site `concept_map` to populate them.

**Usage**

```
toOMOP(scores, concept_map = NULL, person_map = NULL, type_concept_id = 0L)
```

**Arguments**

|                              |                                                                                                                                                                                                                                                                    |
|------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>scores</code>          | A <a href="#">ClinicalScore</a> or a list of them.                                                                                                                                                                                                                 |
| <code>concept_map</code>     | Optional data frame overriding the packaged concept map (columns <code>instrument</code> , <code>subscale</code> , <code>domain_id</code> , <code>measurement_concept_id</code> , <code>unit_concept_id</code> ).                                                  |
| <code>person_map</code>      | Optional named vector/list mapping <code>subject_id</code> to an integer <code>person_id</code> . When absent, a plain positive-integer <code>subject_id</code> (matching " <code>^[0-9]+\$</code> ") is used directly; any other id is left as NA with a warning. |
| <code>type_concept_id</code> | Integer <code>*_type_concept_id</code> for provenance (default <code>0L</code> = "No matching concept"); set to your site's type concept (e.g. EHR).                                                                                                               |

**Value**

A list with MEASUREMENT and OBSERVATION data frames conforming to the OMOP CDM v5.4 column layout.

**References**

OHDSI OMOP CDM v5.4 (MEASUREMENT, OBSERVATION).

**See Also**

[writeOMOPTables\(\)](#), [toFHIRObservation\(\)](#)

**Examples**

```
sc <- methods::new("ClinicalScore", instrument_id = "berg", total = 45,
  subject_id = "1001", timestamp = "2026-07-26T09:00:00Z")
suppressWarnings(toOMOP(sc))
```

---

```
validateFHIRObservation
```

*Validate a FHIR Observation against the bundled JSON schema*

---

### Description

Validate a FHIR Observation against the bundled JSON schema

### Usage

```
validateFHIRObservation(obs)
```

### Arguments

obs                    An "fhir\_observation" (or a list).

### Value

TRUE if valid; otherwise FALSE with the validation errors as an attribute (requires **jsonvalidate**).

### See Also

[toFHIRObservation\(\)](#)

---

```
validateNormativeManifest
```

*Validate a normative-artifact governance manifest*

---

### Description

Checks that a normative reference's manifest carries every required governance field — a semantic version, and non-empty provenance\$source, consent\$status, license\$spdx, governance\$custodian and governance\$access\_level — so an ungoverned artifact cannot be registered or distributed.

### Usage

```
validateNormativeManifest(manifest)
```

### Arguments

manifest              A manifest list, a [GovernedNormativeReference](#), or a path to a manifest.json file.

### Value

TRUE invisibly if valid; otherwise an error naming the missing or invalid fields.

**See Also**

[registerNormative\(\)](#), [GovernedNormativeReference\(\)](#)

**Examples**

```
ref <- GovernedNormativeReference("m", "gait", "gait_speed",
  provenance = list(source = "x"), consent = list(status = "public"),
  license = list(spdx = "CC0-1.0"),
  governance = list(custodian = "lab", access_level = "open"))
validateNormativeManifest(ref)
```

---

|                |                                     |
|----------------|-------------------------------------|
| wmftMedianTime | <i>Median WMFT performance time</i> |
|----------------|-------------------------------------|

---

**Description**

Summarises the Wolf Motor Function Test timed-task performance as the median time across tasks, the conventional WMFT time summary.

**Usage**

```
wmftMedianTime(times)
```

**Arguments**

`times`                Numeric vector of per-task performance times (seconds).

**Value**

The median time (seconds).

**See Also**

[scoreWMFT\(\)](#)

---

|                 |                                                    |
|-----------------|----------------------------------------------------|
| writeFHIRBundle | Write a FHIR Bundle of clinical-score Observations |
|-----------------|----------------------------------------------------|

---

**Description**

Write a FHIR Bundle of clinical-score Observations

**Usage**

```
writeFHIRBundle(scores, path)
```

**Arguments**

|        |                                                                               |
|--------|-------------------------------------------------------------------------------|
| scores | A <a href="#">ClinicalScore</a> , an "fhir_observation", or a list of either. |
| path   | Output .json path.                                                            |

**Value**

path, invisibly.

**See Also**

[toFHIRObservation\(\)](#)

---

|                 |                                    |
|-----------------|------------------------------------|
| writeOMOPTables | Write OMOP CDM tables to CSV files |
|-----------------|------------------------------------|

---

**Description**

Writes each non-empty table from [toOMOP\(\)](#) to <path>/<table>.csv (lower-cased, e.g. measurement.csv), following the one-file-per-table OMOP convention.

**Usage**

```
writeOMOPTables(tables, path)
```

**Arguments**

|        |                                                                                                      |
|--------|------------------------------------------------------------------------------------------------------|
| tables | A <a href="#">ClinicalScore</a> , a list of them, or the list returned by <a href="#">toOMOP()</a> . |
| path   | Output directory (created if needed).                                                                |

**Value**

A character vector of written file paths, invisibly.

**See Also**

[toOMOP\(\)](#)

# Index

applyEquating, 3  
applyEquating(), 10  
as.data.frame.ClinicalScore, 4  
  
classifyResponder, 5  
classifyResponder(), 27, 28  
ClinicalInstrument, 6, 8, 16, 25, 26, 28, 36, 42  
ClinicalInstrument-class, 7  
ClinicalScore, 4, 14, 33–40, 42–49, 51, 53–57, 60  
ClinicalScore (ClinicalScore-class), 8  
ClinicalScore-class, 8  
  
defineGoal, 8, 41  
defineGoal(), 41  
  
equateScores, 10  
equateScores(), 3, 4  
estimateMCID\_anchor, 11  
estimateMCID\_anchor(), 12, 13  
estimateMCID\_distribution, 12  
estimateMCID\_distribution(), 11, 13  
estimateMDC, 13  
estimateMDC(), 6, 11, 12, 28  
  
fromFHIR, 14  
  
gasSummary, 15  
gasSummary(), 41  
getClinimetric, 15  
getClinimetric(), 6, 19  
getInstrument, 14, 16  
getInstrument(), 20, 28, 35–37, 42, 45–47, 51  
getNormative, 17  
getNormative(), 20, 29  
GovernedNormativeReference, 17, 17, 18, 21, 23, 24, 29, 31, 53, 58  
GovernedNormativeReference(), 30, 59  
GovernedNormativeReference-class, 18  
  
listClinimetrics, 19  
listClinimetrics(), 16  
listInstruments, 20  
listInstruments(), 16, 28  
listNormative, 20  
listNormative(), 17, 29  
  
masLevelLabel, 21  
masLevelLabel(), 44  
matchStratum, 21  
matchStratum(), 24  
mdcResponder, 22  
mdcResponder(), 6, 27, 28  
  
normativeDeviation, 23  
normativeDeviation(), 22, 24  
normativeZScore, 23, 23  
normativeZScore(), 22, 23, 30, 31  
  
PhysioAppKit::pcm\_measure(), 25–27  
promisRawToT, 24  
promisRawToT(), 51  
  
raschAnalyze, 25  
raschAnalyze(), 27, 28, 34, 44  
raschRecode, 26  
raschRecode(), 25, 26  
raschResponder, 27  
registerInstrument, 28  
registerInstrument(), 7, 16, 20  
registerNormative, 29  
registerNormative(), 17, 18, 20, 59  
romNormalcy, 29  
romNormalcy(), 30, 31  
romReference, 30  
romReference(), 29, 30  
  
score10MWT, 31  
score10MWT(), 33, 52  
score6MWT, 32  
score6MWT(), 32, 52

scoreARAT, 33  
 scoreARAT(), 40  
 scoreBarthel, 34  
 scoreBarthel(), 26, 43, 44  
 scoreBerg, 34  
 scoreBerg(), 39  
 scoreBPI, 35  
 scoreBPI(), 49  
 scoreCohort, 36  
 scoreEQ5D5L, 37  
 scoreEQ5D5L(), 51  
 scoreFAM, 37  
 scoreFAM(), 38  
 scoreFIM, 38  
 scoreFIM(), 38, 39  
 scoreFMALE, 39  
 scoreFMALE(), 35  
 scoreFMAUE, 39  
 scoreFMAUE(), 33  
 scoreFSS, 40  
 scoreFSS(), 45  
 scoreGAS, 15, 41  
 scoreGAS(), 9, 15  
 scoreInstrument, 42  
 scoreInstrument(), 7, 33–40, 43–49, 51, 53  
 scoreKatz, 43  
 scoreKatz(), 34, 44  
 scoreLawton, 43  
 scoreLawton(), 34, 43  
 scoreMAS, 44  
 scoreMAS(), 21  
 scoreMFIS, 45  
 scoreMFIS(), 40  
 scoreMMSE, 45  
 scoreMMSE(), 47  
 scoreMoCA, 46  
 scoreMoCA(), 46  
 scoreMRCSum, 47  
 scoreMRS, 48  
 scoreMRS(), 49  
 scoreNIHSS, 48  
 scoreNIHSS(), 48  
 scorePainNRS, 49  
 scorePainNRS(), 35  
 scorePROMIS, 50  
 scorePROMIS(), 25  
 scoreSF36, 51  
 scoreTUG, 52  
 scoreTUG(), 32, 33  
 scoreWMFT, 53  
 scoreWMFT(), 59  
 show, ClinicalInstrument-method  
     (ClinicalInstrument-class), 7  
 show, ClinicalScore-method  
     (ClinicalScore-class), 8  
 show, GovernedNormativeReference-method,  
     53  
 summary.responder\_classification  
     (classifyResponder), 5  
  
 toADaM\_ADQS, 54  
 toADaM\_ADQS(), 55  
 toCDISC\_QS, 54  
 toCDISC\_QS(), 36, 54  
 toFHIRObservation, 14, 56  
 toFHIRObservation(), 55, 57, 58, 60  
 toOMOP, 57  
 toOMOP(), 36, 55, 60  
  
 validateFHIRObservation, 58  
 validateFHIRObservation(), 56  
 validateNormativeManifest, 29, 58  
 validateNormativeManifest(), 18  
  
 wmftMedianTime, 59  
 wmftMedianTime(), 53  
 writeFHIRBundle, 14, 60  
 writeFHIRBundle(), 36, 56  
 writeOMOPTables, 60  
 writeOMOPTables(), 57