# Package 'cscore'

July 20, 2025

**Type** Package

**Title** Weighted Composite Scoring for Scales

**Version** 1.0.0

**Author** Hohjin Im

**Maintainer** Hohjin Im <hohjini@uci.edu>

**Description** Weighted composite scoring for scales used in psychological and social behavioral research.

**License** GPL (>= 2)

**Encoding** UTF-8

**Depends** R (>= 4.3.0)

**Imports** dplyr,
tidyr,
purrr,
tibble,
magrittr,
stringr,
psych,
lm.beta,
openxlsx,
ltm,
infotheo,
ranger,
missRanger,
mirt,
glmnet

**LazyData** true

**RoxygenNote** 7.3.2

## Contents

---

`average_score` *Calculate Unweighted Composite Scores*

---

## Description

Create composite scores of scales by specifying the indicators that go into each respective composite variable.

## Usage

```
average_score(
  data = .,
  composite_list,
  digits = 3,
  return_metrics = FALSE,
  file = NULL,
  name = NULL
)
```

## Arguments

| | |
|---|---|
| `data` | A dataframe object. This should be a structured dataset where each column represents a variable and each row represents an observation. |
| `composite_list` | A required `composite_list` object. Each name in the list represents a composite variable, and the corresponding vector contains the column names that are associated with the indicators comprising said composite variable. |
| `digits` | The decimal places for the metrics to be rounded to. Default is 3. This argument is only relevant if `return_metrics = TRUE`. |
| `return_metrics` | Logic to determine whether to return reliability and validity metrics. Set to `TRUE` for a list of dataframes with reliability and validity metrics. |
| `file` | An optional file path. If specified, the results will be written as a formatted excel workbook. This argument is only relevant if `return_metrics = TRUE`. |
| `name` | A required string denoting the name of the composite variable. |

## Details

Unweighted composite scores are calculated as the simple arithmetic mean of the indicators, assigning equal weight to each.

Let $I_{cj}$ denote the value of indicator $j$ for case $c$, and let there be $m$ total indicators. The unweighted composite score for case $c$ is:

$$\bar{C}_c = \frac{1}{m} \sum_{j=1}^{m} I_{cj}$$

where $\bar{C}_c$ is the composite score and all indicators contribute equally regardless of their variance, scale, or distribution.

## Value

If `return_metrics = FALSE`, a dataframe identical to the input dataframe, with additional columns appended at the end, is returned. These new columns represent the calculated composite scores. If `return_metrics = TRUE`, a list containing the following dataframes is returned:

- **Data**: A dataframe with the composite variables appended as new variables.
- **Metrics**: A matrix of indicator loadings and weights metrics.
- **Validity**: A matrix of composite reliability and validity metrics.

## Examples

```
data(grit)

# Specify the named list with composite names and their respective indicators
composite_list <- composite_list(

  # Lower-order composites
  extraversion          = sprintf("e%01d", seq(10)),
  neuroticism           = sprintf("n%01d", seq(10)),
  agreeableness         = sprintf("a%01d", seq(10)),
  conscientiousness     = sprintf("c%01d", seq(10)),
  openness              = sprintf("o%01d", seq(10)),
  consistency_interest  = sprintf("gs%01d", c(2,3,5,7,8,11)),
  perseverance_effort   = sprintf("gs%01d", c(1,4,6,9,10,12)),

  # Higher-order composites
  grit                  = c("consistency_interest", "perseverance_effort")

 )

# Calculate unweighted composite scores
average_score(data = grit,
              composite_list = composite_list)

# Calculate unweighted composite scores, reliability, & validity
average_score(data = grit,
              composite_list = composite_list,
              digits = 3,
              return_metrics = TRUE,
              file = "composite.xlsx")

unlink("composite.xlsx")
```

composite_list *Composite List*

## Description

A duplicate of base R's `list` function. Create a list of composite variable named vectors by specifying their corresponding indicator items. Composite variables with indicators comprising the names of other composite variables are automatically categorized as higher-order composites. These are separated into its own `higher` list of composite vectors. The remaining composite variables are allocated into the `lower` list of composite vectors.

## Usage

```
composite_list(...)

clist(...)

cl(...)
```

## Arguments

| | |
|---|---|
| `...` | Required named objects. Each vector should include the columns representing the indicator variables. |

## Value

Two lists of lower and higher-order named composite vectors.

## Examples

```
data(grit)

# Specify the named list with composite names and their respective indicators
composite_list <- composite_list(

  # Lower-order composites
  extraversion          = sprintf("e%01d", seq(10)),
  neuroticism           = sprintf("n%01d", seq(10)),
  agreeableness         = sprintf("a%01d", seq(10)),
  conscientiousness     = sprintf("c%01d", seq(10)),
  openness              = sprintf("o%01d", seq(10)),
  consistency_interest  = sprintf("gs%01d", c(2,3,5,7,8,11)),
  perseverance_effort   = sprintf("gs%01d", c(1,4,6,9,10,12)),

  # Higher-order composites
  grit                  = c("consistency_interest", "perseverance_effort")

 )
```

`composite_model` *Build a Composite Structural Model from Link Paths*

## Description

Combines one or more sets of 'link' paths into a unified structural model. Each input must be a data frame generated by the 'link()' function, specifying directed associations between variables. The function merges these links, identifies the outcome variable(s), and computes a valid topological ordering. The model is assumed to be a directed acyclic graph.

## Usage

```
composite_model(...)
```

## Arguments

| | |
|---|---|
| `...` | One or more 'link' path objects created by the 'link()' function. Each must be a data frame with columns 'from' and 'to', representing directed edges in the model. |

## Value

A list with three components:

**paths** A data frame of all combined directed edges.

**outcomes** A character vector of terminal outcome node(s) (i.e., nodes with no outgoing edges).

**order** A character vector of all nodes sorted in topological order, from predictors to outcomes.

## Examples

```
# Define composite structure
composite_list <- composite_list(
  extraversion          = sprintf("e%01d", seq(10)),
  neuroticism           = sprintf("n%01d", seq(10)),
  agreeableness         = sprintf("a%01d", seq(10)),
  conscientiousness     = sprintf("c%01d", seq(10)),
  openness              = sprintf("o%01d", seq(10)),
  consistency_interest  = sprintf("gs%01d", c(2,3,5,7,8,11)),
  perseverance_effort   = sprintf("gs%01d", c(1,4,6,9,10,12)),
  grit                  = c("consistency_interest", "perseverance_effort")
)

# Define model structure
composite_model <- composite_model(
  link(
    from = c("extraversion",
             "neuroticism",
             "agreeableness",
             "conscientiousness",
             "openness"),
    to = c("grit")
  )
)
```

composite_score *Calculate Composite Scores*

## Description

Create composite scores of scales by specifying the indicators that go into each respective composite variable. See help documents average_score, correlation_score, discriminant_score, information_score, median_score, or sd_score for information on how the composite scores for each weighting scheme are calculated.

## Usage

```
composite_score(
  data = .,
  composite_list,
  composite_model = NULL,
 weight = c("correlation", "regression", "average", "sd_upweight", "sd_downweight",
  "median", "median_decay", "median_gauss", "mutual_info", "irt", "glm", "pca"),
  decay_rate = 0.5,
  sigma = 0.5,
  entropy = c("emp", "mm", "shrink", "sg"),
  nmi_method = c("geometric", "average"),
  threshold = 10,
  pred_type = c("glm", "rf"),
  item_type = NULL,
  pmm_k = 5,
  maxiter = 10,
  verbose = 0,
  alpha = 0.5,
  nfolds = 10,
  ntrees = 100,
  importance = c("permutation", "impurity"),
  family = c("gaussian", "binomial", "multinomial", "poisson"),
  return_metrics = FALSE,
  digits = 3,
  file = NULL,
  name = NULL,
  seed = NULL
)

cscore(
  data = .,
  composite_list,
  composite_model = NULL,
 weight = c("correlation", "regression", "average", "sd_upweight", "sd_downweight",
  "median", "median_decay", "median_gauss", "mutual_info", "irt", "glm", "pca"),
  decay_rate = 0.5,
  sigma = 0.5,
  entropy = c("emp", "mm", "shrink", "sg"),
  nmi_method = c("geometric", "average"),
  threshold = 10,
```

```
  pred_type = c("glm", "rf"),
  item_type = NULL,
  pmm_k = 5,
  maxiter = 10,
  verbose = 0,
  alpha = 0.5,
  nfolds = 10,
  ntrees = 100,
  importance = c("permutation", "impurity"),
  family = c("gaussian", "binomial", "multinomial", "poisson"),
  return_metrics = FALSE,
  digits = 3,
  file = NULL,
  name = NULL,
  seed = NULL
)

cs(
  data = .,
  composite_list,
  composite_model = NULL,
 weight = c("correlation", "regression", "average", "sd_upweight", "sd_downweight",
  "median", "median_decay", "median_gauss", "mutual_info", "irt", "glm", "pca"),
  decay_rate = 0.5,
  sigma = 0.5,
  entropy = c("emp", "mm", "shrink", "sg"),
  nmi_method = c("geometric", "average"),
  threshold = 10,
  pred_type = c("glm", "rf"),
  item_type = NULL,
  pmm_k = 5,
  maxiter = 10,
  verbose = 0,
  alpha = 0.5,
  nfolds = 10,
  ntrees = 100,
  importance = c("permutation", "impurity"),
  family = c("gaussian", "binomial", "multinomial", "poisson"),
  return_metrics = FALSE,
  digits = 3,
  file = NULL,
  name = NULL,
  seed = NULL
)
```

## Arguments

| | |
|---|---|
| `data` | A dataframe object. This should be a structured dataset where each column represents a variable and each row represents an observation. |
| `composite_list` | A required `composite_list` object. Each name in the list represents a composite variable, and the corresponding vector contains the column names that are associated with the indicators comprising said composite variable. |

`composite_model`
: An optional `composite_model` object. Combines one or more sets of 'link' paths specifying directed associations between variables. The model is assumed to be a directed acyclic graph.

`weight`
: The weighting schema used to calculate composite scores. Choose from the following families:

**Covariance Family:**

- `"average"`: Unweighted average (straight mean).
- `"correlation"`: Correlation-weighted composite scores.
- `"regression"`: Regression-weighted composite scores.

**Standard Deviation (SD) Family:**

- `"sd_upweight"`: Upweights indicators with higher standard deviation.
- `"sd_downweight"`: Downweights indicators with higher standard deviation.

**Median Family:**

- `"median"`: Uses the median of the indicators as the composite score.
- `"median_decay"`: Applies a decay-weighted distance from the median.
- `"median_gauss"`: Applies a Gaussian-weighted distance from the median.

**Information Family:**

- `"mutual_info"`: Shared mutual information-weighted composite scores.

**Discriminant Family:**

- `"irt"`: Item response theory discriminant parameter-weighted composite scores.
- `c("pca", "glm")`: Principal component analysis and generalized linear model weighted composite scores.

`decay_rate`
: Numeric. Reflects the decay rate (i.e., sensitivity) of the distance-to-median weighting schema. The default value is set to 0.5.

`sigma`
: Numeric. Reflects the sigma value for the Gaussian function in the distance-to-median weighting schema. The default value is set to 0.5.

`entropy`
: String. Reflects the mutual information entropy estimator from the `infotheo` package. Four estimators are available: `emp` to compute the entropy of the empirical probability distribution. Empirical entropy is suitable for simple calculations without corrections. `mm` applies an asymptotic bias-corrected estimator making it suitable for small sample sizes. `shrink` applies a shrinkage estimate of the Dirichlet probability distribution to provide a stable estimate useful for small sample sizes or sparse data. `sg` applies a Schurmann-Grassberger estimate of the Dirichlet probability distribution to serve as an alternative to the Shrinkage approach.

`nmi_method`
: String. Reflects the method used for calculating Normalized Mutual Information (NMI) values. `"average"` will normalize MI values using the average entropies of variables A and B. `"geometric"` will normalize MI values using the geometric mean of entropies of variables A and B.

`threshold`
: Integer. Specifies the maximum number of unique values to consider the input as discrete. Defaults to 10.

`pred_type`
: Prediction method schema for predictive weighting. Schemas include `c("glm", "rf")`. `"glm"` runs `glmnet::cv.glmnet()` for regularization linear regression. `"rf"` runs `ranger::ranger()` for Random Forest modeling. Default is `"glm"`.

item_type — String. Specifies the item response model(s) to be used, passed to the itemtype argument in mirt::mirt(). See ?mirt::mirt for the full list of supported item types and their definitions. Defaults to NULL.

pmm_k — Integer. Number of donor candidates used for predictive mean matching (PMM) during imputation. If set to 0, PMM is disabled and raw predictions are used.

maxiter — Integer. Maximum number of iterations for chained equations in missRanger::missRanger(). Iteration continues until convergence or the specified limit is reached.

verbose — Logical. If 1, print progress messages and intermediate output. Defaults to 0.

alpha — Numeric. Elastic net mixing parameter, with 0 <= alpha <= 1. Controls the relative weighting of L1 (lasso) and L2 (ridge) penalties in the model. The penalty term is defined as (1 - alpha)/2 * sum(beta^2) + alpha * sum(abs(beta)), where beta is the vector of coefficients. When alpha = 1, the penalty is pure lasso; when alpha = 0, it is ridge regression.

nfolds — Integer. The number of folds used for random forest k-fold cross-validation. Controls how the data are partitioned during resampling. Must be at least 2. Default is 10.

ntrees — Integer. Number of trees to grow in the random forest model. Default in ranger is 500.

importance — Character string. Specifies the type of variable importance measure to compute. Must be one of "permutation" or "impurity".

- "permutation": Computes mean decrease in predictive accuracy by permuting each variable and measuring the resulting drop in model performance (typically using out-of-bag data). This method is more computationally expensive but generally provides more reliable and differentiated estimates of variable importance. It is robust to scale and less biased in the presence of correlated or irrelevant predictors.
- "impurity": Computes mean decrease in node impurity (e.g., Gini for classification, variance for regression) aggregated over all trees. This method is fast and computed during model training, but can be biased toward variables with many categories or continuous values, and tends to distribute importance more evenly across predictors—even when some are irrelevant or redundant.

Recommend using "impurity" for speed and "permutation" for interpretability, reliability, or feature selection. Default is "permutation".

family — Character string. Specifies the model family. Must be one of "gaussian", "binomial", "multinomial", or "poisson". This argument determines the loss function used in penalized model fitting via cv.glmnet. The supported options are:

- "gaussian": for linear regression with continuous outcomes.
- "binomial": for binary classification using logistic regression. The response should be a factor with two levels or a numeric vector containing 0 and 1.
- "multinomial": for multi-class classification with three or more unordered outcome categories. The response should be a factor.
- "poisson": for modeling count data under a Poisson distribution with a log link. The response should be a non-negative count-valued numeric vector.

Default is "gaussian".

return_metrics — Logic. Determines whether to return reliability and validity metrics. Set to TRUE for a list of dataframes with reliability and validity metrics.

| | |
|---|---|
| digits | Integer. The decimal places for the metrics to be rounded to. Default is 3. This argument is only relevant if return_metrics = TRUE. |
| file | An optional file path. If specified, the results will be written as a formatted excel workbook. This argument is only relevant if return_metrics = TRUE. |
| name | An optional string denoting the name of the study/analysis. |
| seed | Integer. Seed for reproducibility. Defaults to NULL, where no seed is set. |

## Value

If return_metrics = FALSE, a dataframe identical to the input dataframe, with additional columns appended at the end, is returned. These new columns represent the calculated composite scores. If return_metrics = TRUE, a list containing the following dataframes is returned:

- **Data**: A dataframe with the composite variables appended as new variables.
- **Metrics**: A matrix of indicator loadings and weights metrics.
- **Validity**: A matrix of composite reliability and validity metrics.

## Examples

```
data(grit)

# Specify the named list with composite names and their respective indicators
composite_list <- composite_list(

  # Lower-order composites
  extraversion          = sprintf("e%01d", seq(10)),
  neuroticism           = sprintf("n%01d", seq(10)),
  agreeableness         = sprintf("a%01d", seq(10)),
  conscientiousness     = sprintf("c%01d", seq(10)),
  openness              = sprintf("o%01d", seq(10)),
  consistency_interest  = sprintf("gs%01d", c(2,3,5,7,8,11)),
  perseverance_effort   = sprintf("gs%01d", c(1,4,6,9,10,12)),

  # Higher-order composites
  grit                  = c("consistency_interest", "perseverance_effort")

 )

# Calculate unweighted composite scores
composite_score(data = grit,
                composite_list = composite_list,
                weight = "average")
```

---

# correlation_score *Calculate Correlation-Weighted Composite Scores*

---

## Description

Create composite scores of scales by specifying the indicators that go into each respective composite variable.

## Usage

```
correlation_score(
  data = .,
  composite_list,
  weight = c("correlation", "regression"),
  digits = 3,
  return_metrics = FALSE,
  file = NULL,
  name = NULL
)
```

## Arguments

| | |
|---|---|
| `data` | A dataframe object. This should be a structured dataset where each column represents a variable and each row represents an observation. |
| `composite_list` | A required `composite_list` object. Each name in the list represents a composite variable, and the corresponding vector contains the column names that are associated with the indicators comprising said composite variable. |
| `weight` | Required weighting schema. Schemas include `c("correlation", "regression")`. Default is `"correlation"`. |
| `digits` | The decimal places for the metrics to be rounded to. Default is 3. This argument is only relevant if `return_metrics = TRUE`. |
| `return_metrics` | Logic to determine whether to return reliability and validity metrics. Set to `TRUE` for a list of dataframes with reliability and validity metrics. |
| `file` | An optional file path. If specified, the results will be written as a formatted excel workbook. This argument is only relevant if `return_metrics = TRUE`. |
| `name` | A required string denoting the name of the composite variable. |

## Details

Composite scores are calculated as either **correlation-weighted** or **regression-weighted** means of the indicators.

*Correlation-weighted scores.* For a given composite, a correlation matrix is computed using all its indicators. Let $w_j$ denote the weight for indicator $j$, computed as the average correlation with all other indicators:

$$w_j = \frac{1}{n} \sum_{i=1}^{n} r_{ij}, \quad i \neq j$$

where $r_{ij}$ is the Pearson correlation between indicators $i$ and $j$, and $n$ is the number of indicators. Self-correlations ($r_{jj}$) are excluded. The weights are then normalized:

$$w_j^* = \frac{w_j}{\frac{1}{m} \sum_{k=1}^{m} w_k}$$

where $m$ is the number of indicators in the composite. The correlation-weighted composite score $\bar{C}_c$ for case $c$ is then:

$$\bar{C}_c = \frac{1}{m} \sum_{j=1}^{m} I_{cj} \cdot w_j^*$$

where $I_{cj}$ is the value of indicator $j$ for case $c$.

*Regression-weighted scores.* A linear model is fit with the correlation-weighted composite scores as the outcome and the indicators as predictors:

$$\bar{C}_c = \beta_0 + \sum_{j=1}^{m} \beta_j I_{cj} + \varepsilon_c$$

The regression weights are the standardized coefficients $\beta_j$, normalized as:

$$w_j = \frac{\beta_j}{\frac{1}{m}\sum_{k=1}^{m} \beta_k}$$

The regression-weighted composite score is then:

$$\bar{C}_c = \frac{1}{m} \sum_{j=1}^{m} I_{cj} \cdot w_j$$

## Value

If `return_metrics = FALSE`, a dataframe identical to the input dataframe, with additional columns appended at the end, is returned. These new columns represent the calculated composite scores. If `return_metrics = TRUE`, a list containing the following dataframes is returned:

- **Data**: A dataframe with the composite variables appended as new variables.
- **Metrics**: A matrix of indicator loadings and weights metrics.
- **Validity**: A matrix of composite reliability and validity metrics.

## Examples

```
data(grit)

# Specify the named list with composite names and their respective indicators
composite_list <- composite_list(

  # Lower-order composites
  extraversion          = sprintf("e%01d", seq(10)),
  neuroticism           = sprintf("n%01d", seq(10)),
  agreeableness         = sprintf("a%01d", seq(10)),
  conscientiousness     = sprintf("c%01d", seq(10)),
  openness              = sprintf("o%01d", seq(10)),
  consistency_interest  = sprintf("gs%01d", c(2,3,5,7,8,11)),
  perseverance_effort   = sprintf("gs%01d", c(1,4,6,9,10,12)),

  # Higher-order composites
  grit                  = c("consistency_interest", "perseverance_effort")

 )

# Calculate correlation-weighted composite scores
correlation_score(data = grit,
                  composite_list = composite_list)

# Calculate correlation-weighted composite scores, reliability, & validity
```

```
correlation_score(data = grit,
                  composite_list = composite_list,
                  digits = 3,
                  return_metrics = TRUE,
                  file = "composite.xlsx")

unlink("composite.xlsx")
```

---

`discriminant_score` *Calculate Discriminant-Weighted Composite Scores*

---

## Description

Create composite scores of scales by specifying the indicators that go into each respective composite variable.

## Usage

```
discriminant_score(
  data = .,
  composite_list,
  composite_model = NULL,
  weight = c("irt", "pca", "glm"),
  pred_type = c("glm", "rf"),
  item_type = NULL,
  pmm_k = 5,
  maxiter = 10,
  verbose = 0,
  digits = 3,
  alpha = 0.5,
  nfolds = 10,
  ntrees = 100,
  importance = c("permutation", "impurity"),
  family = c("gaussian", "binomial", "multinomial", "poisson"),
  return_metrics = FALSE,
  seed = NULL,
  file = NULL,
  name = NULL
)
```

## Arguments

| | |
|---|---|
| `data` | A dataframe object. This should be a structured dataset where each column represents a variable and each row represents an observation. |
| `composite_list` | A required `composite_list` object. Each name in the list represents a composite variable, and the corresponding vector contains the column names that are associated with the indicators comprising said composite variable. |
| `composite_model` | An optional `composite_model` object. Combines one or more sets of 'link' paths specifying directed associations between variables. The model is assumed to be a directed acyclic graph. |

weight — Required weighting schema. Schemas include c("irt","pca", "glm"). c("pca", "glm") run the same weighting schema. Default is "irt".

pred_type — Prediction method schema for predictive weighting. Schemas include c("glm", "rf"). "glm" runs glmnet::cv.glmnet() for regularization linear regression. "rf" runs ranger::ranger() for Random Forest modeling. Default is "glm".

item_type — Character string or vector specifying the item response model(s) to be used, passed to the itemtype argument in mirt::mirt(). See ?mirt::mirt for the full list of supported item types and their definitions. Defaults to NULL.

pmm_k — Integer. Number of donor candidates used for predictive mean matching (PMM) during imputation. If set to 0, PMM is disabled and raw predictions are used.

maxiter — Integer. Maximum number of iterations for chained equations in missRanger::missRanger(). Iteration continues until convergence or the specified limit is reached.

verbose — Logical; if 1, print progress messages and intermediate output. Defaults to 0.

digits — The decimal places for the metrics to be rounded to. Default is 3. This argument is only relevant if return_metrics = TRUE.

alpha — Elastic net mixing parameter, with 0 <= alpha <= 1. Controls the relative weighting of L1 (lasso) and L2 (ridge) penalties in the model. The penalty term is defined as (1 - alpha)/2 * sum(beta^2) + alpha * sum(abs(beta)), where beta is the vector of coefficients. When alpha = 1, the penalty is pure lasso; when alpha = 0, it is ridge regression.

nfolds — Integer. The number of folds used for random forest k-fold cross-validation. Controls how the data are partitioned during resampling. Must be at least 2. Default is 10.

ntrees — Integer. Number of trees to grow in the random forest model. Default in ranger is 500.

importance — Character string specifying the type of variable importance measure to compute. Must be one of "permutation" or "impurity".

- "permutation": Computes mean decrease in predictive accuracy by permuting each variable and measuring the resulting drop in model performance (typically using out-of-bag data). This method is more computationally expensive but generally provides more reliable and differentiated estimates of variable importance. It is robust to scale and less biased in the presence of correlated or irrelevant predictors.
- "impurity": Computes mean decrease in node impurity (e.g., Gini for classification, variance for regression) aggregated over all trees. This method is fast and computed during model training, but can be biased toward variables with many categories or continuous values, and tends to distribute importance more evenly across predictors—even when some are irrelevant or redundant.

Recommend using "impurity" for speed and "permutation" for interpretability, reliability, or feature selection. Default is "permutation".

family — Character string specifying the model family. Must be one of "gaussian", "binomial", "multinomial", or "poisson". This argument determines the loss function used in penalized model fitting via cv.glmnet. The supported options are:

- "gaussian": for linear regression with continuous outcomes.
- "binomial": for binary classification using logistic regression. The response should be a factor with two levels or a numeric vector containing 0 and 1.

- `"multinomial"`: for multi-class classification with three or more unordered outcome categories. The response should be a factor.
- `"poisson"`: for modeling count data under a Poisson distribution with a log link. The response should be a non-negative count-valued numeric vector.

Default is `"gaussian"`.

`return_metrics` Logic to determine whether to return reliability and validity metrics. Set to `TRUE` for a list of dataframes with reliability and validity metrics.

`seed` An integer seed for reproducibility. Defaults to `NULL`, where no seed is set.

`file` An optional file path. If specified, the results will be written as a formatted excel workbook. This argument is only relevant if `return_metrics = TRUE`.

`name` A required string denoting the name of the composite variable.

## Details

Discriminant composite scores combine psychometric discrimination and predictive utility into a single weighting schema. Two primary discriminant sources are supported: **latent composite loadings** and **predictive weights**. The composite score for case $c$ is computed as:

$$\bar{C}_c = \frac{1}{m} \sum_{j=1}^{m} I_{cj} \cdot w_j^*$$

where $I_{cj}$ is the value of indicator $j$ for case $c$, and $w_j^*$ is the final normalized weight. The weighting proceeds in two stages:

**1. Discriminant weighting from latent structure.**

If `weight = "irt"`, a unidimensional Item Response Theory (IRT) model is estimated:

$$P(Y_{cj} = 1) = \text{logit}^{-1}(a_j \cdot \theta_c + b_j)$$

where $a_j$ is the discrimination parameter for indicator $j$, and $\theta_c$ is the latent trait estimate for case $c$. The discrimination parameters $a_j$ are extracted and normalized:

$$w_j = \frac{a_j}{\sum_{k=1}^{m} a_k}$$

If `weight = "glm"` or `"pca"`, a single-factor principal component analysis (PCA) is conducted. The first factor score is extracted as a latent proxy $Z_c$. A penalized linear model is then fit:

$$Z_c = \sum_{j=1}^{m} \beta_j \cdot I_{cj} + \varepsilon_c$$

where $\beta_j$ are estimated using elastic net regularization (via **glmnet**). These are normalized analogously to IRT:

$$w_j = \frac{\beta_j}{\sum_{k=1}^{m} \beta_k}$$

**2. Predictive weighting from external outcomes (optional).**

If one or more outcome variables are provided via `outcomes`, an additional predictive weighting stage is applied. Two methods are available:

*(a) Generalized Linear Model (GLM).* Each outcome $Y$ is regressed on the indicators $I_{cj}$ using elastic net regularization. For each fitted model, the normalized coefficients $\gamma_j^{(r)}$ are extracted. The final predictive weights are averaged over all outcomes:

$$p_j = \frac{1}{R} \sum_{r=1}^{R} \gamma_j^{(r)}$$

where $R$ is the number of outcomes. The predictive weights are then normalized:

$$p_j^* = \frac{p_j}{\sum_{k=1}^{m} p_k}$$

*(b) Random Forest (RF).* Each outcome $Y$ is predicted via a random forest, and variable importance scores (e.g., permutation or impurity) are extracted for each indicator. These are averaged across outcomes and normalized in the same manner as the GLM approach.

**3. Final weight aggregation.**

If predictive weights $p_j^*$ are available, they are combined with the latent discrimination weights $w_j$ to form final composite weights:

$$w_j^* = \frac{w_j + p_j^*}{\frac{1}{m} \sum_{k=1}^{m} (w_k + p_k^*)}$$

If no predictive targets are provided, $w_j^*$ defaults to normalized discrimination weights.

## Value

If `return_metrics = FALSE`, a dataframe identical to the input dataframe, with additional columns appended at the end, is returned. These new columns represent the calculated composite scores. If `return_metrics = TRUE`, a list containing the following dataframes is returned:

- **Data**: A dataframe with the composite variables appended as new variables.
- **Metrics**: A matrix of indicator loadings and weights metrics.
- **Validity**: A matrix of composite reliability and validity metrics.

## Examples

```
data(grit)

# Specify the named list with composite names and their respective indicators
composite_list <- composite_list(

  # Lower-order composites
  extraversion          = sprintf("e%01d", seq(10)),
  neuroticism           = sprintf("n%01d", seq(10)),
  agreeableness         = sprintf("a%01d", seq(10)),
  conscientiousness     = sprintf("c%01d", seq(10)),
  openness              = sprintf("o%01d", seq(10)),
  consistency_interest  = sprintf("gs%01d", c(2,3,5,7,8,11)),
  perseverance_effort   = sprintf("gs%01d", c(1,4,6,9,10,12)),

  # Higher-order composites
  grit                  = c("consistency_interest", "perseverance_effort")
```

```
 )

# Calculate discriminant-weighted composite scores
discriminant_score(data = grit,
                   composite_list = composite_list)

# Calculate discriminant-weighted composite scores, reliability, & validity
discriminant_score(data = grit,
                   composite_list = composite_list,
                   digits = 3,
                   return_metrics = TRUE,
                   file = "composite.xlsx")

unlink("composite.xlsx")
```

---

`export_metrics` *Export Metrics (Reliability & Validity)*

---

## Description

A function to export a `metrics` object as a formatted excel workbook to the specified `file` path. The format is consistent with what is typically reported in APA 7th edition, and can be copy and pasted directly into a Word document or similar document software.

## Usage

```
export_metrics(metrics, digits, name = NULL, file)
```

## Arguments

| | |
|---|---|
| `metrics` | A required list object. The list should contain an array for composite scores under `data`, a data frame of indicator loadings and weights under `metrics`, and composite validity value under `validity`. |
| `digits` | The decimal places for the metrics to be rounded to. Default is 3. |
| `name` | An optional string denoting the study/analysis name. |
| `file` | A required file path. If specified, the results will be written as a formatted excel workbook. |

## Details

Indicator loadings ($\lambda_i$) are computed as the bivariate correlation between each indicator $I_i$ and the correlation-weighted composite score $\bar{C}_c$:

$$\lambda_i = \text{cor}(I_i, \bar{C}_c)$$

Two reliability metrics are reported: Cronbach's alpha ($\alpha$) and composite reliability ($\rho_c$, also referred to as McDonald's omega, $\omega$).

Cronbach's alpha is defined as:

$$\alpha = \frac{k\bar{c}}{\bar{v} + (k-1)\bar{c}}$$

where $k$ is the number of indicators, $\bar{c}$ is the average inter-item covariance, and $\bar{v}$ is the average indicator variance. Alpha assumes tau-equivalence (equal loadings across indicators) and is sensitive to the number of indicators. While widely used, it is not recommended due to these limitations; it is reported here for completeness only.

Composite reliability ($\rho_c$) is a weighted reliability estimate based on user-defined indicator weights. It uses the same formula as McDonald's omega but allows loadings to be weighted according to the composite score structure:

$$\rho_c = \frac{\left(\sum \lambda_i w_i\right)^2}{\left(\sum \lambda_i w_i\right)^2 + \sum e_{w_i}}$$

where $w_i$ is the weight for indicator $i$, and $e_{w_i}$ is the weighted measurement error:

$$e_{w_i} = (1 - \lambda_i^2) \cdot w_i$$

Although this metric is labeled $\rho_c$ to align with PLS-SEM nomenclature, it is mathematically equivalent to omega and may be reported as either.

Average Variance Extracted (AVE) measures the proportion of variance captured by the construct relative to measurement error. It is defined as:

$$\text{AVE} = \frac{\sum \lambda_{w_i}^2}{\sum \lambda_{w_i}^2 + \sum e_{w_i}}$$

where the weighted squared loading is:

$$\lambda_{w_i}^2 = \lambda_i^2 \cdot w_i$$

AVE is commonly used in the Fornell-Larcker criterion for discriminant validity:

$$\sqrt{\text{AVE}_i} > \max_j(r_{ij})$$

That is, the square root of AVE for a construct should exceed its highest correlation with any other construct.

## Value

A formatted excel workbook saved to specified file path.

`grit` *Grit Data*

## Description

This example data contains 500 responses from people completing the Big 5 personality inventory (John & Srivastava, 1999) and the Grit Scale (Duckworth et al., 2007) from OpenPsychoMetrics.

## Usage

```
data(grit)
```

## Format

A data frame with 500 rows and 63 variables. All items are scored on a 1–5 Likert scale. Items marked with an asterisk (*) are reverse-coded and have been recoded for convenience:

`id` Respondent ID

**Extraversion:**

`e1` I am the life of the party.

`e2` I don't talk a lot. (*))

`e3` I feel comfortable around people.

`e4` I keep in the background. (*)

`e5` I start conversations.

`e6` I have little to say. (*)

`e7` I talk to a lot of different people at parties.

`e8` I don't like to draw attention to myself. (*)

`e9` I don't mind being the center of attention.

`e10` I am quiet around strangers. (*)

**Neuroticism:**

`n1` I get stressed out easily.

`n2` I am relaxed most of the time. (*)

`n3` I worry about things.

`n4` I seldom feel blue. (*)

`n5` I am easily disturbed.

`n6` I get upset easily.

`n7` I change my mood a lot.

`n8` I have frequent mood swings.

`n9` I get irritated easily.

`n10` I often feel blue.

**Agreeableness:**

`a1` I feel little concern for others. (*)

`a2` I am interested in people.

`a3` I insult people. (*)

`a4` I sympathize with others' feelings.

`a5` I am not interested in other people's problems. (*)

`a6` I have a soft heart.

`a7` I am not really interested in others. (*)

`a8` I take time out for others.

`a9` I feel others' emotions.

`a10` I make people feel at ease.

**Conscientiousness:**

`c1` I am always prepared.

`c2` I leave my belongings around. (*)

`c3` I pay attention to details.

`c4` I make a mess of things. (*)

`c5` I get chores done right away.

`c6` I often forget to put things back in their proper place. (*)

`c7` I like order.

`c8` I shirk my duties. (*)

`c9` I follow a schedule.

`c10` I am exacting in my work.

**Openness:**

`o1` I have a rich vocabulary.

`o2` I have difficulty understanding abstract ideas. (*)

`o3` I have a vivid imagination.

`o4` I am not interested in abstract ideas. (*)

`o5` I have excellent ideas.

`o6` I do not have a good imagination. (*)

`o7` I am quick to understand things.

`o8` I use difficult words.

`o9` I spend time reflecting on things.

`o10` I am full of ideas.

**Grit:**

`gs1` I have overcome setbacks to conquer an important challenge.

`gs2` New ideas and projects sometimes distract me from previous ones. (*)

`gs3` My interests change from year to year. (*)

`gs4` Setbacks don't discourage me.

`gs5` I have been obsessed with a certain idea or project for a short time but later lost interest. (*)

`gs6` I am a hard worker.

`gs7` I often set a goal but later choose to pursue a different one. (*)

`gs8` I have difficulty maintaining my focus on projects that take more than a few months to complete. (*)

`gs9` I finish whatever I begin.

`gs10` I have achieved a goal that took years of work.

`gs11` I become interested in new pursuits every few months. (*)

`gs12` I am diligent.

### Source

OpenPsychoMetrics.

### References

Duckworth, A.L., Peterson, C., Matthews, M.D., & Kelly, D.R. (2007). Grit: Perseverance and passion for long-term goals. Journal of Personality and Social Psychology, 9, 1087-1101.

John, O. P., & Srivastava, S. (1999). The Big-Five trait taxonomy: History, measurement, and theoretical perspectives. In L. A. Pervin & O. P. John (Eds.), Handbook of personality: Theory and research (2nd ed., pp. 102–138). Guilford Press.

---

## `information_score` *Calculate Information-Weighted Composite Scores*

---

### Description

Create composite scores of scales by specifying the indicators that go into each respective composite variable.

### Usage

```
information_score(
  data = .,
  composite_list,
  entropy = c("emp", "mm", "shrink", "sg"),
  nmi_method = c("geometric", "average"),
  threshold = 10,
  digits = 3,
  return_metrics = FALSE,
  file = NULL,
  name = NULL
)
```

### Arguments

| | |
|---|---|
| `data` | A dataframe object. This should be a structured dataset where each column represents a variable and each row represents an observation. |
| `composite_list` | A required `composite_list` object. Each name in the list represents a composite variable, and the corresponding vector contains the column names that are associated with the indicators comprising said composite variable. |
| `entropy` | A string value reflecting the mutual information entropy estimator from the `infotheo` package. Four estimators are available: `emp` to compute the entropy of the empirical probability distribution. Empirical entropy is suitable for simple calculations without corrections. `mm` applies an asymptotic bias-corrected estimator making it suitable for small sample sizes. `shrink` applies a shrinkage estimate of the Dirichlet probability distribution to provide a stable estimate useful for small sample sizes or sparse data. `sg` applies a Schurmann-Grassberger estimate of the Dirichlet probability distribution to serve as an alternative to the Shrinkage approach. |

| | |
|---|---|
| `nmi_method` | A string value reflecting the method used for calculating Normalized Mutual Information (NMI) values. `"average"` will normalize MI values using the average entropies of variables A and B. `"geometric"` will normalize MI values using the geometric mean of entropies of variables A and B. |
| `threshold` | An integer specifying the maximum number of unique values to consider the input as discrete. Defaults to 10. |
| `digits` | The decimal places for the metrics to be rounded to. Default is 3. This argument is only relevant if `return_metrics = TRUE`. |
| `return_metrics` | Logic to determine whether to return reliability and validity metrics. Set to `TRUE` for a list of dataframes with reliability and validity metrics. |
| `file` | An optional file path. If specified, the results will be written as a formatted excel workbook. This argument is only relevant if `return_metrics = TRUE`. |
| `name` | A required string denoting the name of the composite variable. |

## Details

Composite scores are computed as the **information-weighted** mean of the indicators.

*Information-weighted scores.* For a given composite, pairwise mutual information (MI) is computed between all indicators to form an information matrix $I_{ij}$. Self-information values on the diagonal are excluded. Each mutual information value is then normalized to obtain the Normalized Mutual Information (NMI).

The NMI between variables $i$ and $j$ can be computed using either arithmetic mean:

$$\mathrm{NMI}(i,j) = \frac{2 \cdot I(i,j)}{H(i) + H(j)}$$

or geometric mean:

$$\mathrm{NMI}(i,j) = \frac{I(i,j)}{\sqrt{H(i) \cdot H(j)}}$$

where $I(i,j)$ is the mutual information and $H(i)$ and $H(j)$ are the entropies of variables $i$ and $j$, respectively. The method is selected via `nmi_method`. The information weight for indicator $j$ is the average of its NMI values:

$$w_j = \frac{1}{n} \sum_{i \neq j} \mathrm{NMI}(i,j)$$

where $n$ is the number of indicators and the sum excludes self-pairs. The weights are then normalized:

$$w_j^* = \frac{w_j}{\frac{1}{m}\sum_{k=1}^{m} w_k}$$

where $m$ is the number of indicators in the composite. The information-weighted composite score $\bar{C}_c^{\mathrm{MI}}$ for case $c$ is calculated as:

$$\bar{C}_c^{\mathrm{MI}} = \frac{1}{m} \sum_{j=1}^{m} I_{cj} \cdot w_j^*$$

where $I_{cj}$ is the value of indicator $j$ for case $c$.

## Value

If `return_metrics = FALSE`, a dataframe identical to the input dataframe, with additional columns appended at the end, is returned. These new columns represent the calculated composite scores. If `return_metrics = TRUE`, a list containing the following dataframes is returned:

- **Data**: A dataframe with the composite variables appended as new variables.
- **Metrics**: A matrix of indicator loadings and weights metrics.
- **Validity**: A matrix of composite reliability and validity metrics.

## Examples

```
data(grit)

# Specify the named list with composite names and their respective indicators
composite_list <- composite_list(

  # Lower-order composites
  extraversion          = sprintf("e%01d", seq(10)),
  neuroticism           = sprintf("n%01d", seq(10)),
  agreeableness         = sprintf("a%01d", seq(10)),
  conscientiousness     = sprintf("c%01d", seq(10)),
  openness              = sprintf("o%01d", seq(10)),
  consistency_interest  = sprintf("gs%01d", c(2,3,5,7,8,11)),
  perseverance_effort   = sprintf("gs%01d", c(1,4,6,9,10,12)),

  # Higher-order composites
  grit                  = c("consistency_interest", "perseverance_effort")

 )

# Calculate correlation-weighted composite scores
information_score(data = grit,
                  composite_list = composite_list)

# Calculate correlation-weighted composite scores, reliability, & validity
information_score(data = grit,
                  composite_list = composite_list,
                  digits = 3,
                  return_metrics = TRUE,
                  file = "composite.xlsx")

unlink("composite.xlsx")
```

---

# link *Create Link Paths Between Node Sets*

---

## Description

Generates a set of directed paths representing all combinations of links from nodes in ‘from‘ to nodes in ‘to‘. The result is a data frame that can be used as input to ‘composite_model()‘ and related functions.

### Usage

```
link(from, to)
```

### Arguments

| | |
|---|---|
| from | A character vector of source node names. |
| to | A character vector of target node names. |

### Value

A data frame with two columns, 'from' and 'to', containing all directed edges from each element in 'from' to each element in 'to'.

### Examples

```
link(from = c("x1", "x2"), to = c("y"))
```

---

## median_score *Calculate Median Composite Scores & Metrics*

---

### Description

Create composite scores of scales by specifying the indicators that go into each respective composite variable.

### Usage

```
median_score(
  data = .,
  composite_list,
  weight = c("median", "median_decay", "median_gauss"),
  decay_rate = 0.5,
  sigma = 0.5,
  digits = 3,
  return_metrics = FALSE,
  file = NULL,
  name = NULL
)
```

### Arguments

| | |
|---|---|
| data | A dataframe object. This should be a structured dataset where each column represents a variable and each row represents an observation. |
| composite_list | A required composite_list object. Each name in the list represents a composite variable, and the corresponding vector contains the column names that are associated with the indicators comprising said composite variable. |
| weight | Required weighting schema. Schemas include c("median","median_decay", "median_gauss"). Default is "median". |

| | |
|---|---|
| `decay_rate` | A numeric value reflecting the decay rate (i.e., sensitivity) of the distance-to-median weighting schema. The default value is set to 0.5. |
| `sigma` | A numeric value reflecting the sigma value for the Gaussian function in the distance-to-median weighting schema. The default value is set to 0.5. |
| `digits` | The decimal places for the metrics to be rounded to. Default is 3. This argument is only relevant if `return_metrics` = `TRUE`. |
| `return_metrics` | Logic to determine whether to return reliability and validity metrics. Set to `TRUE` for a list of dataframes with reliability and validity metrics. |
| `file` | An optional file path. If specified, the results will be written as a formatted excel workbook. This argument is only relevant if `return_metrics` = `TRUE`. |
| `name` | A required string denoting the name of the composite variable. |

## Details

Composite scores can be calculated using three median-based weighting schemes: **unweighted median**, **decay-weighted**, and **Gaussian-weighted**.

*Unweighted median.* The median composite score $\bar{C}_c^{\text{med}}$ for case $c$ is:

$$\bar{C}_c^{\text{med}} = \text{median}(I_{c1}, I_{c2}, \ldots, I_{cm})$$

where $I_{cj}$ is the value of indicator $j$ for case $c$, and $m$ is the number of indicators.

*Decay-weighted median.* For each respondent, let $M_c$ denote the within-row median:

$$M_c = \text{median}(I_{c1}, I_{c2}, \ldots, I_{cm})$$

The absolute distance to the median is:

$$D_{cj} = |I_{cj} - M_c|$$

The decay weight for indicator $j$ is then:

$$w_{cj} = \exp(-\gamma \cdot D_{cj})$$

where $\gamma$ is the user-specified `decay_rate`. Weights are row-normalized:

$$w_{cj}^* = \frac{w_{cj}}{\frac{1}{m}\sum_{k=1}^{m} w_{ck}}$$

The decay-weighted median composite score is:

$$\bar{C}_c^{\text{decay}} = \frac{1}{m}\sum_{j=1}^{m} I_{cj} \cdot w_{cj}^*$$

*Gaussian-weighted median.* Using the same distance $D_{cj}$, the Gaussian weight is:

$$w_{cj} = \exp\left(-\frac{D_{cj}^2}{2\sigma^2}\right)$$

where $\sigma$ is the user-specified `sigma` parameter controlling the spread. Weights are row-normalized as before, and the final score is:

$$\bar{C}_c^{\text{gauss}} = \frac{1}{m} \sum_{j=1}^{m} I_{cj} \cdot w_{cj}^*$$

All median-based methods treat each case independently. Weighting is performed per respondent, allowing differential emphasis on indicators based on their proximity to the respondent-specific median.

## Value

If `return_metrics = FALSE`, a dataframe identical to the input dataframe, with additional columns appended at the end, is returned. These new columns represent the calculated composite scores. If `return_metrics = TRUE`, a list containing the following dataframes is returned:

- **Data**: A dataframe with the composite variables appended as new variables.
- **Metrics**: A matrix of indicator loadings and weights metrics.
- **Validity**: A matrix of composite reliability and validity metrics.

## Examples

```
data(grit)

# Specify the named list with composite names and their respective indicators
composite_list <- composite_list(

  # Lower-order composites
  extraversion          = sprintf("e%01d", seq(10)),
  neuroticism           = sprintf("n%01d", seq(10)),
  agreeableness         = sprintf("a%01d", seq(10)),
  conscientiousness     = sprintf("c%01d", seq(10)),
  openness              = sprintf("o%01d", seq(10)),
  consistency_interest  = sprintf("gs%01d", c(2,3,5,7,8,11)),
  perseverance_effort   = sprintf("gs%01d", c(1,4,6,9,10,12)),

  # Higher-order composites
  grit                  = c("consistency_interest", "perseverance_effort")

 )

# Calculate median composite scores
median_score(data = grit,
             composite_list = composite_list)

# Calculate median composite scores, reliability, & validity
median_score(data = grit,
             composite_list = composite_list,
             digits = 3,
             return_metrics = TRUE,
             file = "composite.xlsx")

unlink("composite.xlsx")
```

sd_score *Calculate Standard Deviation Composite Scores*

## Description

Create composite scores of scales by specifying the indicators that go into each respective composite variable.

## Usage

```
sd_score(
  data = .,
  composite_list,
  weight = c("sd_upweight", "sd_downweight"),
  digits = 3,
  return_metrics = FALSE,
  file = NULL,
  name = NULL
)
```

## Arguments

| | |
|---|---|
| data | A dataframe object. This should be a structured dataset where each column represents a variable and each row represents an observation. |
| composite_list | A required composite_list object. Each name in the list represents a composite variable, and the corresponding vector contains the column names that are associated with the indicators comprising said composite variable. |
| weight | Required weighting schema. Schemas include c("sd_upweight", "sd_downweight"). Default is "sd_upweight". |
| digits | The decimal places for the metrics to be rounded to. Default is 3. This argument is only relevant if return_metrics = TRUE. |
| return_metrics | Logic to determine whether to return reliability and validity metrics. Set to TRUE for a list of dataframes with reliability and validity metrics. |
| file | An optional file path. If specified, the results will be written as a formatted excel workbook. This argument is only relevant if return_metrics = TRUE. |
| name | A required string denoting the name of the composite variable. |

## Details

Standard deviation (SD)–weighted composite scores assign indicator weights based on their empirical variability. Two variants are supported: **upweighting** and **downweighting** by standard deviation.

Let $I_{cj}$ denote the value of indicator $j$ for case $c$. For each indicator $j$, its sample standard deviation $\sigma_j$ is computed as:

$$\sigma_j = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(I_{ij} - \bar{I}_{\cdot j})^2}$$

where $\bar{I}_{\cdot j}$ is the sample mean of indicator $j$ across all $n$ observations.

**1. SD-Upweighted Composite.**

Indicators with greater dispersion are weighted more heavily. The weights are normalized:

$$w_j = \frac{\sigma_j}{\frac{1}{m}\sum_{k=1}^{m}\sigma_k}$$

The final composite score for case $c$ is:

$$\bar{C}_c^{(u)} = \frac{1}{m}\sum_{j=1}^{m} I_{cj} \cdot w_j$$

where $m$ is the total number of indicators.

**2. SD-Downweighted Composite.**

Indicators with greater dispersion are weighted less heavily by using the inverse of the standard deviation:

$$w_j = \frac{\sigma_j^{-1}}{\frac{1}{m}\sum_{k=1}^{m}\sigma_k^{-1}}$$

The resulting downweighted composite score is:

$$\bar{C}_c^{(d)} = \frac{1}{m}\sum_{j=1}^{m} I_{cj} \cdot w_j$$

Both upweighted and downweighted schemes produce unit-normalized weights ($\bar{w} = 1$). These can be used to emphasize or attenuate variability in the indicators depending on the analytic goal.

If `return_metrics = TRUE`, the function returns reliability and validity metrics in addition to the composite score. See `calc_metrics` for details.

## Value

If `return_metrics = FALSE`, a dataframe identical to the input dataframe, with additional columns appended at the end, is returned. These new columns represent the calculated composite scores. If `return_metrics = TRUE`, a list containing the following dataframes is returned:

- **Data**: A dataframe with the composite variables appended as new variables.
- **Metrics**: A matrix of indicator loadings and weights metrics.
- **Validity**: A matrix of composite reliability and validity metrics.

## Examples

```
data(grit)

# Specify the named list with composite names and their respective indicators
composite_list <- composite_list(

  # Lower-order composites
  extraversion          = sprintf("e%01d", seq(10)),
  neuroticism           = sprintf("n%01d", seq(10)),
  agreeableness         = sprintf("a%01d", seq(10)),
```

```
  conscientiousness     = sprintf("c%01d", seq(10)),
  openness              = sprintf("o%01d", seq(10)),
  consistency_interest  = sprintf("gs%01d", c(2,3,5,7,8,11)),
  perseverance_effort   = sprintf("gs%01d", c(1,4,6,9,10,12)),

  # Higher-order composites
  grit                  = c("consistency_interest", "perseverance_effort")

 )

# Calculate correlation-weighted composite scores
sd_score(data = grit,
         composite_list = composite_list)

# Calculate correlation-weighted composite scores, reliability, & validity
sd_score(data = grit,
         composite_list = composite_list,
         digits = 3,
         return_metrics = TRUE,
         file = "composite.xlsx")

unlink("composite.xlsx")
```

# Index