Package: esgcalc (via r-universe)

October 25, 2026

Type Package

Title Offline Environmental, Social and Governance Calculations

Version 0.2.0

Description Native, deterministic calculations for selected environmental, social and governance metrics. Provides granular taxonomy criteria, non-financial taxonomy indicators, principal adverse impact components, portfolio calculations and configurable performance ratings. Versioned local resources, explicit assumptions and structured results support reproducible offline use. Does not determine legal applicability or provide a complete regulatory reporting system. Selected methods reference EU Regulations 2020/852, 2022/1288 and 2023/2772; exact editions and provision locators are supplied in the bundled source contracts. For background on the taxonomy framework, see the European Commission <https://finance.ec.europa.eu/sustainable-finance/tools-and-standards/eu-taxonomy-sustainable-activities_en>.

License GPL (>= 3)

Copyright RiskDataScience GmbH

Encoding UTF-8

Depends R (>= 4.1.0)

Imports jsonlite, digest

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Dimitrios Geromichalos [aut, cre], RiskDataScience GmbH [cph]

Maintainer Dimitrios Geromichalos <riskdatascience@web.de>

Repository https://rds0001.r-universe.dev

Date/Publication 2026-10-24 11:17:45 UTC

RemoteUrl https://github.com/cran/esgcalc

RemoteRef HEAD

RemoteSha 0deec54f6a1a10110e4f28e5699bcd5a8ea49555

Contents

esgcalc-package . . . 3
aggregate_ghg_inventory . . . 4
aggregate_issuer_positions . . . 5
aggregate_pai_observations . . . 6
aggregate_pai_portfolio . . . 7
allocate_multi_objective_contributions . . . 9
apply_rating_limits . . . 10
board_metrics . . . 11
calculate_pai_indicator . . . 13
check_quantitative_criterion . . . 14
combine_taxonomy_assessments . . . 15
convert_currency . . . 17
diversity_metrics . . . 18
emissions_from_activity . . . 19
energy_metrics . . . 20
esg_context . . . 22
esgcalc-contracts . . . 23
esgcalc-resources . . . 27
explain_rating . . . 29
export_resources . . . 30
financed_emissions . . . 31
ghg_intensity . . . 32
health_safety_metrics . . . 33
list_resources . . . 35
load_local_data . . . 36
load_profile . . . 37
load_reference_data . . . 38
metric_change . . . 39
normalize_indicator . . . 40
overall_rating . . . 42
pay_gap . . . 43
pillar_scores . . . 44
portfolio_concentration . . . 45
portfolio_coverage . . . 47
portfolio_exposure . . . 48
portfolio_rating_summary . . . 49
position_weights . . . 50
print.esg_result . . . 51
rate_company . . . 52
rating_sensitivity . . . 53
reconcile_kpi_allocations . . . 55
taxonomy_capex . . . 56
taxonomy_opex . . . 57
taxonomy_turnover . . . 58
validate_activity_assessment . . . 60
validate_governance_findings . . . 61

validate_inputs 62
validate_pai_inputs 63
validate_rating_model 64
waste_resource_metrics 66
water_metrics 67
workforce_metrics 68

Index **70**

`esgcalc-package` *Offline Environmental, Social and Governance Calculations*

Description

Native, deterministic calculations for selected environmental, social and governance metrics, quantitative taxonomy criteria, principal adverse impact components, portfolios and configurable performance ratings.

Details

Inputs are structured observations and explicit external assessments. The package performs calculations locally, without network access or a Python installation. It does not determine legal applicability, investigate evidence or provide complete regulatory reporting.

Start with `vignette("offline-workflow", package = "esgcalc")`. esgcalc-contracts documents input schemas, units, numerical methods and limitations. esgcalc-resources describes bundled reference data, rule profiles and fictional company cases.

Use esg_context for an explicit calculation context. Results contain values, units, statuses and provenance; missing observations are not silently replaced by zero. Malformed inputs raise classified `esg_error` conditions. The ESG rating is a transparent configurable model, not a statutory grade.

Author(s)

Dimitrios Geromichalos. Copyright: RiskDataScience GmbH.

See Also

pay_gap, check_quantitative_criterion, calculate_pai_indicator, rate_company, list_resources

Examples

```
gap <- pay_gap(20, 18)
stopifnot(gap$status == "OK", gap$value == 10)
print(gap)
stopifnot(pay_gap(0, 18)$status == "NOT_APPLICABLE")
model <- validate_rating_model()
stopifnot(isTRUE(model$value))
```

`aggregate_ghg_inventory` *Aggregate the declared scopes; market/location alternatives are never added*

Description

ENV-002: Aggregate the declared scopes; market/location alternatives are never added.

Usage

```
aggregate_ghg_inventory(rows, scope2_method, required_scopes = c("1", "2",
    "3"), context = NULL)
```

Arguments

`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`scope2_method`	location or market. The two Scope 2 alternatives are never added together.
`required_scopes`	Distinct scope identifiers chosen from 1, 2 and 3, as character values.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Aggregate the declared scopes; market/location alternatives are never added. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_ESRS_2023; ESRS E1-6 paragraphs 44-55; AR39-55. Kind: METRIC_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "scope2_alternatives", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(aggregate_ghg_inventory, args)
stopifnot(result$status == "OK")
result$value
```

---

aggregate_issuer_positions

Aggregate position values while requiring consistent duplicated issuer attributes

---

Description

PORT-007: Aggregate position values while requiring consistent duplicated issuer attributes.

Usage

```
aggregate_issuer_positions(rows, context = NULL)
```

Arguments

`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Aggregate position values while requiring consistent duplicated issuer attributes. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_METHODS_V2; CONTRACTS.md portfolio methods. Kind: ANALYTICAL_METHOD.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "issuer", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(aggregate_issuer_positions, args)
stopifnot(result$status == "OK")
result$value
```

---

aggregate_pai_observations

Mean of complete quarter-end observations, or explicitly selected additional observations

---

Description

PAI-002: Mean of complete quarter-end observations, or explicitly selected additional observations.

Usage

```
aggregate_pai_observations(observations, sampling = "quarter_ends", context = NULL)
```

Arguments

`observations`	Named dated rating observations with value, unit, as_of and source, or PAI observation rows with as_of, indicator_id, component, unit, method_id, status and value.
`sampling`	quarter_ends requires exactly four statutory quarter ends; all_observations_equal allows additional dates in that year.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Mean of complete quarter-end observations, or explicitly selected additional observations. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_SFDR_RTS_CONSOLIDATED_20230220; Article 6(3); ESA JC2023_18 Section IV: quarter-end mean, optional additional observations. Kind: LEGAL_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "four_quarters", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(aggregate_pai_observations, args)
stopifnot(result$status == "OK")
result$value
```

---

aggregate_pai_portfolio

Calculate selected PAI components on one unchanged portfolio universe

---

Description

PAI-003: Calculate selected PAI components on one unchanged portfolio universe.

Usage

```
aggregate_pai_portfolio(rows, indicator_ids, averaging = NULL, context = NULL)
```

Arguments

`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`indicator_ids`	Distinct numeric integer IDs from 1 through 18.
`averaging`	value_weighted or issuer_mean where applicable. The ESA2025 v2 profile fixes value_weighted; other indicators reject this option.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Calculate selected PAI components on one unchanged portfolio universe. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Weighted means for 5/6/12/13 are declared method choices; PAI18 PED ambiguity has explicit interpretations See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_SFDR_RTS_CONSOLIDATED_20230220; Annex I Table 1 indicators 1-18 and definitions; ESA JC2023_18 Section IV questions 10,11,17,27,28. Kind: LEGAL_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "pai_portfolio", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(aggregate_pai_portfolio, args)
stopifnot(result$status == "OK")
result$value
```

`allocate_multi_objective_contributions` *Allocate disjoint slices once in total; each slice may contribute to multiple goals*

Description

TAX-004: Allocate disjoint slices once in total; each slice may contribute to multiple goals.

Usage

```
allocate_multi_objective_contributions(amounts, allocations, context = NULL)
```

Arguments

`amounts`	Rows containing unique amount_id and nonnegative amount in context currency.
`allocations`	Disjoint rows with unique allocation_id, amount_id and fraction in 0..1; taxonomy contributions additionally require distinct goal codes.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Allocate disjoint slices once in total; each slice may contribute to multiple goals. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_TAX_KPI_CONSOLIDATED_20260101; Annex I 1.2.2; Annex II: avoid double counting. Kind: LEGAL_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "multi_goal_union", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(allocate_multi_objective_contributions, args)
stopifnot(result$status == "OK")
result$value
```

`apply_rating_limits` *Apply most restrictive triggered cap; unknown required findings suppress the score*

Description

RATE-004: Apply most restrictive triggered cap; unknown required findings suppress the score.

Usage

```
apply_rating_limits(score, findings, model = NULL, context = NULL)
```

Arguments

`score`	Normalized score in 0..100; NULL is an unavailable score.
`findings`	Named logical findings for rating limits, or governance finding rows with finding_id, assessment and optional topic. Missing required evidence remains unknown.
`model`	Named model list; NULL loads rating_reference.json. Includes normative origin, entity type, positive E/S/G weights, per-pillar indicator weights, units, piecewise knots, coverage threshold, maximum age, limits and descending classes.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Apply most restrictive triggered cap; unknown required findings suppress the score. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Normative performance model, not a statutory rating or empirically calibrated financial-risk model See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_REFERENCE_MODEL_V1; RATING.md; rating_reference.json. Kind: MODEL_CHOICE.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "cap", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(apply_rating_limits, args)
stopifnot(result$status == "OK")
result$value
```

---

board_metrics — *Female representation uses all board members, including declared other/unknown gender*

Description

GOV-001: Female representation uses all board members, including declared other/unknown gender.

Usage

```
board_metrics(female, male, other = 0, unknown_gender = 0, independent = NULL,
    context = NULL)
```

Arguments

`female`	Nonnegative integer count. Optional NULL means unknown, never zero. Subset and population consistency rules are enforced.
`male`	Nonnegative integer count. Optional NULL means unknown, never zero. Subset and population consistency rules are enforced.
`other`	Nonnegative integer count. Optional NULL means unknown, never zero. Subset and population consistency rules are enforced.
`unknown_gender`	Nonnegative integer count. Optional NULL means unknown, never zero. Subset and population consistency rules are enforced.

`independent` Nonnegative integer count. Optional NULL means unknown, never zero. Subset and population consistency rules are enforced.

`context` Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Female representation uses all board members, including declared other/unknown gender. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Female/all-member denominator follows SFDR; do not label it ESRS female/male ratio See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_ESRS_2023; ESRS 2 GOV-1 paragraph21; SFDR Annex I Table1 indicator13. Kind: METRIC_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "board", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(board_metrics, args)
stopifnot(result$status == "OK")
result$value
```

calculate_pai_indicator

Calculate an enumerated Table 1 component; averaging conventions must be explicitly selected

Description

PAI-001: Calculate an enumerated Table 1 component; averaging conventions must be explicitly selected.

Usage

```
calculate_pai_indicator(indicator_id, rows, averaging = NULL, context = NULL)
```

Arguments

`indicator_id`	Registered indicator identifier: integer 1..18 for PAI, or the exact rating model indicator name.
`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`averaging`	value_weighted or issuer_mean where applicable. The ESA2025 v2 profile fixes value_weighted; other indicators reject this option.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Calculate an enumerated Table 1 component; averaging conventions must be explicitly selected. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Weighted means for 5/6/12/13 are declared method choices; PAI18 PED ambiguity has explicit interpretations See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_SFDR_RTS_CONSOLIDATED_20230220; Annex I Table 1 indicators 1-18 and definitions; ESA JC2023_18 Section IV questions 10,11,17,27,28. Kind: LEGAL_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "pai_attribution", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(calculate_pai_indicator, args)
stopifnot(result$status == "OK")
result$value
```

---

check_quantitative_criterion

Compute a selected clause from the edition-pinned numerical criterion catalog

---

Description

TAX-002: Compute a selected clause from the edition-pinned numerical criterion catalog.

Usage

```
check_quantitative_criterion(criterion_id, values, context = NULL)
```

Arguments

`criterion_id`	Exact ID from the 23-clause criteria.json catalog. Unsupported IDs or editions are rejected.
`values`	Named structured inputs in the units specified by the selected criterion in criteria.json; optional units mapping must agree exactly.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Compute a selected clause from the edition-pinned numerical criterion catalog. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_TAX_CLIMATE_CONSOLIDATED_20260101; Per-clause locators in criteria.json; circular clauses use EU_TAX_ENV_CONSOLIDATED_20260101. Kind: LEGAL_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "ped_boundary", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(check_quantitative_criterion, args)
stopifnot(result$status == "OK")
result$value
```

---

combine_taxonomy_assessments

Combine explicitly sourced assessments for six goals; never infer missing evidence

---

Description

TAX-001: Combine explicitly sourced assessments for six goals; never infer missing evidence.

Usage

```
combine_taxonomy_assessments(eligible, substantial_contribution, dnsh, minimum_safeguards,
    technical_screening, context = NULL)
```

Arguments

`eligible`	Sourced eligibility assessment: status, origin, source, method, scope, evidence and assessed_on.
`substantial_contribution`	Named assessments keyed by CCM, CCA, WTR, CE, PPC and BIO. Omitted goals remain unknown.
`dnsh`	Named do-no-significant-harm assessments for the six goals. NOT_APPLICABLE requires external justification.

`minimum_safeguards`
: Sourced minimum-safeguards assessment. NOT_APPLICABLE cannot waive this requirement.

`technical_screening`
: Named technical-screening assessments for the six goals. Claimed goals cannot waive screening.

`context`
: Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Combine explicitly sourced assessments for six goals; never infer missing evidence. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_TAX_BASE_2020; Articles 3, 9, 17, 18. Kind: LEGAL_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "tax_alignment", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(combine_taxonomy_assessments, args)
stopifnot(result$status == "OK")
result$value
```

`convert_currency` *Use explicit target-currency units per one source-currency unit; never fetch or invert*

Description

PORT-005: Use explicit target-currency units per one source-currency unit; never fetch or invert.

Usage

```
convert_currency(amount, source_currency, target_currency, rate, rate_date,
    as_of, rate_source, context = NULL)
```

Arguments

`amount`	Finite signed currency amount; conversion uses the supplied direct rate.
`source_currency`	Three-uppercase-letter source currency code.
`target_currency`	Three-uppercase-letter target currency code.
`rate`	Positive target-currency units per source-currency unit; same-currency conversion requires one.
`rate_date`	ISO calendar date of the supplied rate; must equal as_of.
`as_of`	Explicit ISO knowledge or conversion date.
`rate_source`	Nonempty provenance for the supplied exchange rate.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Use explicit target-currency units per one source-currency unit; never fetch or invert. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_METHODS_V2; CONTRACTS.md portfolio methods. Kind: ANALYTICAL_METHOD.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "fx", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(convert_currency, args)
stopifnot(result$status == "OK")
result$value
```

---

diversity_metrics — *Compute disjoint category shares including an explicit unknown category where needed*

Description

SOC-002: Compute disjoint category shares including an explicit unknown category where needed.

Usage

```
diversity_metrics(groups, context = NULL)
```

Arguments

groups	Disjoint workforce rows (group_id, headcount, fte) or diversity rows (category, count). No inferred population overlap.
context	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Compute disjoint category shares including an explicit unknown category where needed. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_ESRS_2023; ESRS S1-9 paragraphs64-66; caller defines exhaustive population. Kind: METRIC_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "diversity", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(diversity_metrics, args)
stopifnot(result$status == "OK")
result$value
```

`emissions_from_activity` *Multiply explicit activity quantities by sourced factors, output tonnes CO2e*

Description

ENV-001: Multiply explicit activity quantities by sourced factors, output tonnes CO2e.

Usage

```
emissions_from_activity(rows, context = NULL)
```

Arguments

`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Multiply explicit activity quantities by sourced factors, output tonnes CO2e. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

GHG_CORPORATE_REVISED_2004; Chapter 6: identifying and calculating GHG emissions. Kind: ACCOUNTING_METHOD.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "activity_emissions", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(emissions_from_activity, args)
stopifnot(result$status == "OK")
result$value
```

---

energy_metrics — *Separate consumption/production and compute high-impact-sector revenue intensity*

Description

ENV-004: Separate consumption/production and compute high-impact-sector revenue intensity.

Usage

```
energy_metrics(rows, revenues = list(), context = NULL)
```

Arguments

`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`revenues`	Revenue rows with sector, net_revenue and currency; matched to high-impact consumption sectors.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Separate consumption/production and compute high-impact-sector revenue intensity. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_ESRS_2023; ESRS E1-5 paragraphs 37-43; AR32-38. Kind: METRIC_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "energy", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(energy_metrics, args)
stopifnot(result$status == "OK")
result$value
```

esg_context — *Declare an explicit calculation context*

Description

Select a pinned method profile, knowledge date, reporting period, currency and calculation mode. Selection does not determine legal applicability.

Usage

```
esg_context(profile_id = "esgcalc-methods-v1", as_of = NULL,
  period_start = NULL, period_end = NULL, currency = "EUR",
  mode = "REFERENCE", scenario_reason = NULL)
```

Arguments

`profile_id`	Exact local profile identifier.
`as_of`	ISO knowledge date or scalar Date; required by dated legal profiles.
`period_start`, `period_end`	ISO reporting dates or scalar Date values. A start requires an end; the end cannot exceed the knowledge date.
`currency`	Three uppercase letters; no implicit currency conversion.
`mode`	REFERENCE or SCENARIO.
`scenario_reason`	Nonempty explanation required for SCENARIO.

Details

Missing dates remain unspecified. Invalid dates, reversed periods and unsupported calculation modes raise esg_error conditions. Unsupported profile applicability is checked when calculating.

Value

An esg_context named list, revalidated at every calculation.

Examples

```
ctx <- esg_context(as_of = "2026-09-30")
stopifnot(ctx$currency == "EUR")
tryCatch(esg_context(as_of = "2026-02-30"),
  esg_error = function(e) stopifnot(e$code == "INVALID_DATE"))
```

esgcalc-contracts *Structured input schemas and numerical method contracts*

Description

Complete row schemas, units, missing-value policy and algorithm boundaries inherited from the pinned Python Golden Source.

Common conventions

Public functions return `esg_result`: `value`, `unit`, `status`, `function_id`, `profile_id`, `source_ids`, `method_id`, `assessment`, `mode`, `input_hash`, `details`, `limitations`. Results are native named R lists. `esg_error` exposes `code`, `field`, message. Malformed inputs raise; well-formed missing observations return `MISSING_INPUT`, `PARTIAL` or `UNRATED`. `NOT_APPLICABLE` is distinct from a zero result. Never coerce NULL to zero.

Use finite JSON numbers (not booleans), ISO dates, explicit string IDs and unique row IDs. Units are exact, case-sensitive identifiers. Unless `percent` is stated, proportions use 0..1. Scores use 0..100; higher means stronger performance according to the selected model. No decision-time rounding. Amounts are nonnegative net rows; signed accounting corrections must be reconciled before input. Money is in currency units, never implicit millions.

The context is revalidated at each call:

```
esg_context(profile_id = "esgcalc-methods-v1",
            as_of = NULL, period_start = NULL, period_end = NULL,
            currency = "EUR", mode = "REFERENCE",
            scenario_reason = NULL)
```

Scenarios require a reason. Legal profiles require explicit dates within their declared interval: `eu-taxonomy-2025-v1` (taxonomy only), `eu-taxonomy-2026-v1` (taxonomy/KPI), `sfdr-pai-v1` (2023–2026). Profiles never determine the caller's legal applicability. They expose bounded methods, not an attestation of complete/current regulation.

External assessment schema: `{status, origin, evidence, source, method, scope, assessed_on, reason?}`. All fields except reason are required; reason is required for NOT_APPLICABLE. Assessments after the knowledge date are rejected. Status is MET / NOT_MET / UNKNOWN / NOT_APPLICABLE; origin is COMPUTED, EXTERNAL_ASSESSMENT or DECLARED_INPUT. Evidence is a nonempty source reference. NOT_APPLICABLE requires a reason. Metadata is structurally checked; neither source truth nor assessor competence is independently established.

Taxonomy

Goal codes: CCM, CCA, WTR, CE, PPC, BIO. `combine_taxonomy_assessments` takes eligibility, minimum safeguards and goal-keyed SC, DNSH, technical-screening assessments. A goal requires eligibility, SC, technical screening, safeguards and DNSH for the other five goals. Any known failure dominates unknowns; a single satisfied goal is sufficient for the combined result. Missing goals are unknown. Validated NOT_APPLICABLE DNSH is an external scope decision,

not an automatic exemption. `validate_activity_assessment` adds unique activity_id rows with eligible/sc/dnsh/minimum_safeguards/technical_screening fields.

`check_quantitative_criterion` computes the 23 explicitly enumerated numerical clauses in '`TAXONOMY_CRITERIA.md`' in the installed '`doc`' directory. That catalog specifies all field names, fixed units, formulas, boundaries and exact edition locators. The 2025/legacy profiles support only the two unchanged original clauses (new-building PED and light-vehicle manufacturing CO2). The expanded catalog is pinned to the 2026 consolidated editions. Unsupported criteria are rejected. A passed numerical clause is not full activity alignment; the other named conditions, DNSH and minimum safeguards remain separate inputs.

Allocation input: amounts `{amount_id, amount}`; allocations `{allocation_id, amount_id, fraction, goals:[...]}`. Slices are disjoint and fractions for one amount cannot exceed one. Each slice counts once in the union, while contributing to multiple goal totals. Goal totals are not additive.

Nonfinancial taxonomy KPIs

Every row: amount_id, amount, currency, period_end, accounting_category, eligible and alignment assessments. Currency must equal Context; row period_end equals Context.period_end (or as_of when omitted). as_of is the knowledge/review date, not automatically the reporting date. Input amounts already represent disjoint economic slices; allocation_fraction, if present, must be one. `reconcile_kpi_allocations` checks fractions separately.

Turnover includes net_revenue, excludes other_income. Adapted-only turnover is excluded unless adaptation_enabling or otherwise_aligned is explicitly true. CapEx includes IAS16, IAS38, IAS40_FAIR, IAS40_COST, IAS41, IFRS16_ROU, GAAP_EQUIVALENT additions; excludes fair_value_change, depreciation, non_rou_lease. National GAAP equivalence and the financial boundary are caller assessments. OpEx includes research_development, building_renovation, short_lease, maintenance_repair, day_to_day_servicing, gaap_noncapitalised_lease. The last requires accounting_basis=NATIONAL_GAAP_NO_ROU. capitalised, other_cost and already_in_capex=true are excluded. Categories use the regulatory direct-cost boundary; library users must provide correctly mapped accounting records.

CapEx/OpEx numerator_category: aligned (default), plan or measure. Plan: purpose=expand_aligned / upgrade_eligible, start/end, approved/disclosed assessments. Duration <=5 years; >5 and <=10 requires extended_justification. After end, completion_aligned is required. When criteria_amended_on applies, updated_on must be within two years and resets the start; overdue updates fail. Unknown pending updates suppress the complete ratio. `taxonomy_capex` accepts prior_periods: each record has period_end, optional period_start, rows and previously_reported_aligned_numerator. Historical plan rows identify plan_id; current rows contain the matching plan with plan_id and current review. The function recalculates historical numerators using those reviews and reports restated_periods, numerator_adjustment, plan_failures and restatement_required. It does not change the caller's accounting ledger. Every plan requiring a historical review must be supplied; missing reviews are an error. Measure: measure_start, operational_on, measure_qualifies assessment. Completion may be planned within 18 months; the date is not proof of actual implementation.

non_material=true invokes the explicit 2026 unassessed option; combined omitted amount must be strictly below 10% of the respective KPI denominator. This remains in the denominator and is separately reported. OpEx `immaterial_reason` suppresses both eligibility/alignment output while retaining its denominator. Unknown non-omitted assessments suppress the quote; known aligned amounts remain visible. Rows may identify taxonomy_activity: 4.26–4.28 nuclear, 4.29–4.31 fossil gas, or another explicit activity code/other. A complete supplied breakdown produces separate

aligned/eligible amounts and denominator shares; if any activity code is absent, nuclear_and_gas is NULL, never a guessed zero.

`eu-taxonomy-legacy-fy2025-v2` implements the Article 4 election in 2026/73: period_start must be in 2025, period_end explicit; the new 10% unassessed option is unavailable. Applicability/election are chosen externally. The 2026 profile is a frozen edition, not a promise to incorporate future amendments automatically. Financial institutions' GAR and specialist reporting are outside this profile.

PAI Table 1 numerical contract

All rows include position_id, issuer_id, value (EUR), currency=EUR, as_of and asset_class=company / sovereign / real_estate / cash / other. Include the entire declared portfolio; do not silently filter cash or unknown assets to improve coverage. Money inputs EVIC/revenue/GDP are EUR; emissions tCO2e, water/waste tonnes. Company scope2 is one already-selected compatible measurement, never the sum of location and market alternatives. Caller supplies measurement boundaries.

| Indicator | Additional input | Calculation | |—|—|—| | 1 | evic, scope1/2/3 | Sum value/EVIC * emissions, each scope and total | | 2 | same | Attributed total divided by all investments in million EUR | | 3 | scope1/2/3, revenue_eur | Investment-weighted emissions per million EUR revenue | | 4 | fossil_fuel boolean | Positive company exposure / all investments | | 5 | nonrenewable_consumption_mwh, consumption_mwh, nonrenewable_production_mwh, production_mwh | Two separate average fractions | | 6 | sector (NACE section), energy_gwh, revenue_eur | Average GWh/million EUR revenue per section A–H or L | | 7 | biodiversity_adverse boolean | Positive company exposure / all investments | | 8 | evic, water_emissions_t | Attributed tonnes / all investments in million EUR | | 9 | evic, hazardous_radioactive_waste_t | Same attribution with waste tonnes | | 10 | ungc_oecd_violation boolean | Positive company exposure / all investments | | 11 | missing_compliance_process boolean | Positive company exposure / all investments | | 12 | male_hourly_pay, female_hourly_pay | Average signed (male-female)/male *100 | | 13 | female_board, total_board | Average female/all-members fraction | | 14 | controversial_weapons boolean | Positive company exposure / all investments | | 15 | country_emissions_tco2e, gdp_eur | Investment-weighted emissions/million EUR GDP | | 16 | country_id, social_violation boolean | Distinct violating countries and fraction of all investee countries | | 17 | fossil_real_estate boolean | Positive real-estate exposure / all real-estate investments | | 18 | energy_inefficient, epc_nzeb_applicable booleans | Inefficient exposure / applicable real-estate exposure |

The recommended `sfdr-pai-2023-esa2025-v2` profile fixes value-weighted applicable-company means for 5/6/12/13 (per sector for 6). This is a declared method convention where the legal text says average, not a claimed uniquely mandated weighting. The older `sfdr-pai-v1` analytical interface requires an explicit value_weighted or issuer_mean selection. Its presence is not evidence that every alternative is accepted for statutory reporting.

The v2 profile requires valuation_basis=FISCAL_YEAR_END_PRICES, valuation_date, fx_date, fiscal_year_end (all equal), and lookthrough_complete=true. Holdings are already resolved to underlying positions. Optional position_units, unit_price, unit_price_currency and fx_rate (EUR per pricing currency) derive the value; a separately supplied inconsistent value is rejected. Optional EVIC components common_equity_market_value, preferred_equity_market_value, book_debt and noncontrolling_interests derive EVIC without subtracting cash. All four must be supplied together. EUR conversion of company financial statements remains an explicit upstream step, supported by convert_currency.

Missing required fields suppress an indicator rather than deleting a position. EVIC/revenue/GDP

must be positive. PAI5 zero total consumption/production with zero corresponding nonrenewable quantity is NOT_APPLICABLE for that component; positive nonrenewable quantity over zero total is invalid. Zero-production companies are excluded only from that component's applicable mean. Company attributes must agree for repeated issuers; real-estate asset_id, when supplied, is the classification key so distinct buildings of one owner can differ.

PAI18 accepts either explicit external booleans or structured building inputs: epc_nzeb_applicable; built_on; for older buildings epc A..G, with C..G inefficient; for newer buildings ped, nzeb_threshold, unit=kWh/m2/year, ped_interpretation. The archived corrected formula literally uses PED *below* NZEB. Therefore RTS_LITERAL_BELOW follows that wording. Reversing it requires PERFORMANCE_NONCOMPLIANCE_SCENARIO and Context.mode=SCENARIO. There is no silent interpretation. Exactly 2020-12-31 requires cutoff_policy=year_end_inclusive. Unknown applicability is missing; explicitly inapplicable assets are outside the PAI18 denominator. See '`METHOD_ISSUES.md`' in the installed '`doc`' directory.

`aggregate_pai_observations` requires March31, June30, September30 and December31 of one year, not later than the knowledge date. Each scalar observation has as_of, indicator_id, component, unit, method_id and value; metadata must agree. The default averages exactly four observations. Additional dates require sampling=all_observations_equal and all mandatory quarter ends remain necessary. Missing values suppress the mean; a ratio of pooled yearly numerators is not a substitute for the mean of reference-date ratios. Multi-component indicators are split into separate component series explicitly. `aggregate_pai_portfolio` calculates the selected indicators on the same full investment universe.

Relevant optional Table2/3 positions are mapped, with exact coverage and remaining steps, in optional_pai_catalog.json. Reusable company metrics and exposure functions are not advertised as a complete optional-PAI statement. Product classification, reporting materiality decisions and entity disclosures are outside this calculation engine.

Environment, social and governance

Emissions rows: activity_id, quantity, activity_unit, factor, factor_activity_unit, factor_emission_unit, factor_source. Multiplication converts to tCO2e. Factors already include the chosen GWP boundary; there is no factor lookup or estimation. Inventory rows: emission_id, scope=1 / 2_location / 2_market / 3, value, unit. Explicitly choose scope2_method and required_scopes. Absent required scopes are missing. Units: kgCO2e/tCO2e; kg/t; kWh/MWh/GWh/MJ/GJ; litre/m3; m/km; unit. Cross-dimensional conversion is rejected. Intensity inputs are emissions in tCO2e and an explicitly named denominator. Zero denominator returns NOT_APPLICABLE. Energy rows: energy_id, category=renewable/fossil/nuclear, value, unit, flow=consumption (default) or production. Production and consumption never sum into one denominator. Optional revenues rows {sector, net_revenue, currency} plus energy sector codes enable A–H/L consumption divided by corresponding revenue. Every high-impact revenue sector needs an explicit energy row (including zero), and every high-impact energy sector needs its revenue. Water: withdrawn/discharged/reused in one unit on a common boundary; consumption =withdrawn-discharged, own analytical reuse share=reused/(withdrawn+reused). Alternatively supply facilities {facility_id, withdrawn, discharged, reused, unit, water_risk}; this also calculates water-risk-area consumption. Unknown risk does not silently become false. stored_start/stored_end report storage and change separately; net_revenue_eur enables m3/million EUR intensity. Storage is not deducted twice from consumption. Waste rows: waste_id, value, unit, destination=reused / recycled / other_recovery / disposal, hazardous boolean. Material rows: material_id, value, unit, secondary boolean, optional reused boolean (a subset of

secondary). Recycling, reuse/recycling and recovery shares are distinct. Change requires comparable=true; relative change only for a positive baseline. Optional restatement={restated_baseline, reason} retains both original and restated baseline.

Workforce rows: group_id, headcount, fte, disjoint groups. Diversity rows: category, count, disjoint and exhaustive including unknown if relevant. Pay gap uses gross hourly male/female averages and returns signed percent. Turnover uses leavers / turnover_denominator; denominator_method is mandatory when turnover is requested. Workforce groups are disjoint and the turnover population must be comparable. No individual employee records are required. Health/safety: recordable_injuries, hours_worked, optional fatalities, fatalities_from_ill_health, recordable_ill_health_cases, days_lost. Missing optional counts remain NULL. Injury and illness fatalities are separate subsets. Rates use one million hours; severity-days rate is an analytical convention. Optional covered_workers, total_workers, audited_covered_workers produce management-system coverage. Day-count convention must match the intended disclosure (ESRS/SFDR workday and calendar-day populations are not interchangeable). Board denominator includes female, male, other, unknown_gender; independent is a separate optional count. Governance findings are finding_id + assessment records. Conflicting states for one supplied topic and scope are rejected. Definitions are selected metrics, not the complete ESRS disclosure requirements.

Portfolios and local data

Positions: position_id, issuer_id, value>=0, currency, as_of if Context is dated. No leverage/shorts, look-through or market-data retrieval. Weights normalize over all positions; coverage counts non-missing field values; exposure requires a boolean field. HHI is sum of squared grouped weights. FX rate is target currency units per source unit, with matching explicit dates and rate_source. Financed emissions accepts listed_equity/market_value and listed_company_bond/book_value, positive evic, emissions_tco2e and emissions_scope. One compatible emission boundary is required. No other PCAF asset classes or quality scores are asserted. Issuer grouping requires equal issuer_data values.

`validate_inputs` takes rows and schema={unique_key?, fields:{name:{type, required?, minimum?, maximum?, enum?, unit_field?, unit?, schema?}}}; types number/integer/text/boolean/date/object/array. Nested object/array schemas describe mapping rows recursively. additional_fields=allow (default) or reject is explicit. Missing required values produce a missing report. `load_local_data` accepts JSON or CSV; without schema CSV stays strings. With schema, declared numeric and true/false fields are parsed and validated; no guessed units/dates or nested CSV encoding. UTF-8 BOM is supported; nonfinite JSON, duplicate headers and inconsistent row widths are rejected. Resource names are manifest-registered JSON basenames; hashes are checked. Export requires an explicit local destination and rejects existing targets. No implicit file writes, networking, secrets, global caches or automatic resource updates occur.

R hashes use sorted canonical JSON and SHA-256. Numeric rendering differs between languages: input and model hashes are stable within R, not guaranteed identical to Python. Resource file SHA-256 values are byte-identical.

`esgcalc-resources` *Versioned offline catalogs and fictional example data*

Description

Fifteen UTF-8 JSON files installed under extdata: a manifest plus fourteen registered resources copied unchanged from the Python Golden Source.

Details

Use `list_resources()` to inspect the manifest and `load_reference_data(name)` to verify a registered file's SHA-256 before reading it. Data are returned as named lists with arrays represented by lists; no global objects are installed.

criteria.json Twenty-three selected quantitative taxonomy clauses, input units, thresholds, formulas, remaining conditions and source locators.

source_contracts.json Source or method contracts for all 47 capability IDs.

profiles.json Explicit method profiles and bounded date/domain applicability.

source_status.json Frozen source-edition and review status.

pai_catalog.json Mandatory Table 1 numerical component metadata.

optional_pai_catalog.json Optional indicator inventory; inclusion in this catalog is not an implementation claim.

rating_reference.json Normative six-indicator E/S/G model, scales, weights, age and coverage rules, limits and classes.

reference_cases.json Forty-two independent arithmetic/reference cases with kwargs, context, expected status and dotted-path checks.

synthetic_realistic_cases.json One hundred and twenty fictional annual cases for sixty companies across six archetypes and two years. Raw physical, workforce, financial, governance and assessment inputs are reconciled.

synthetic_realistic_expected.json Independently derived Decimal expectations for the realistic cases; not generated by the calculation runtime.

synthetic_companies.json, synthetic_operations.json, synthetic_portfolio.json, synthetic_scenarios.json Earlier fictional regression datasets, operations, portfolios and scenarios.

All companies and measurements are fictional. External classification and evidence fields are declared assumptions. Data are not empirical calibration evidence. Registered resource files may be exported explicitly using `export_resources()`; the package never writes into its installation.

See Also

load_reference_data, list_resources, export_resources

Examples

```
resources <- list_resources()$value
length(resources$files)
model <- load_reference_data("rating_reference.json")$value
model$id
```

`explain_rating` *Return stored rating contributions and reasons, without inventing narrative conclusions*

Description

RATE-007: Return stored rating contributions and reasons, without inventing narrative conclusions.

Usage

```
explain_rating(rating, context = NULL)
```

Arguments

`rating`	An esg_result produced by rate_company(), or its equivalent named list.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Return stored rating contributions and reasons, without inventing narrative conclusions. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Normative performance model, not a statutory rating or empirically calibrated financial-risk model See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_REFERENCE_MODEL_V1; RATING.md; rating_reference.json. Kind: MODEL_CHOICE.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
c <- Filter(function(x) x$id == "company", cases)[[1]]
r <- do.call(rate_company, c(c$kwargs, list(context = do.call(esg_context, c$context))))
explain_rating(r)
```

export_resources *Export selected resources to an explicit local directory; never overwrite existing files*

Description

DATA-006: Export selected resources to an explicit local directory; never overwrite existing files.

Usage

```
export_resources(destination, names, context = NULL)
```

Arguments

destination	Explicit local export directory. Existing target files are never overwritten.
names	Distinct registered resource basenames, supplied as a character vector or list.
context	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Export selected resources to an explicit local directory; never overwrite existing files. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_DATA_CONTRACT_V2; CONTRACTS.md; API.md. Kind: INPUT_CONTRACT.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
d <- tempfile("esgcalc-export-")
export_resources(d, names = "profiles.json")
stopifnot(file.exists(file.path(d, "profiles.json")))
unlink(d, recursive = TRUE)
```

`financed_emissions` *Declared listed-equity/listed-company-bond attribution using positive EVIC and sourced emissions*

Description

PORT-006: Declared listed-equity/listed-company-bond attribution using positive EVIC and sourced emissions.

Usage

```
financed_emissions(rows, context = NULL)
```

Arguments

`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Declared listed-equity/listed-company-bond attribution using positive EVIC and sourced emissions. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Listed-company equities at market value and bonds at book value; EVIC, no cash subtraction; other PCAF asset classes excluded See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

PCAF_PART_A_2025; Section5.1 listed equity and corporate bonds. Kind: ACCOUNTING_METHOD.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "financed", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(financed_emissions, args)
stopifnot(result$status == "OK")
result$value
```

---

ghg_intensity — *Divide tonnes CO2e by an explicitly named positive physical/economic basis*

Description

ENV-003: Divide tonnes CO2e by an explicitly named positive physical/economic basis.

Usage

```
ghg_intensity(emissions, denominator, denominator_unit, context = NULL)
```

Arguments

`emissions`	Nonnegative emissions in tCO2e, or NULL for missing data.
`denominator`	Nonnegative scalar denominator; zero produces NOT_APPLICABLE.
`denominator_unit`	Explicit unit label for the intensity denominator; no implicit millions conversion.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Divide tonnes CO2e by an explicitly named positive physical/economic basis. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_ESRS_2023; ESRS E1-6 paragraphs 44-55; AR39-55. Kind: METRIC_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "intensity", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(ghg_intensity, args)
stopifnot(result$status == "OK")
result$value
```

---

health_safety_metrics *Accident frequency, separately classified fatalities, days lost and coverage*

Description

SOC-004: Accident frequency, separately classified fatalities, days lost and coverage.

Usage

```
health_safety_metrics(recordable_injuries, hours_worked, fatalities = NULL,
    fatalities_from_ill_health = NULL, recordable_ill_health_cases = NULL,
    days_lost = NULL, covered_workers = NULL, total_workers = NULL,
    audited_covered_workers = NULL, context = NULL)
```

Arguments

`recordable_injuries`
: Nonnegative integer count. Optional NULL means unknown, never zero. Subset and population consistency rules are enforced.

`hours_worked`
: Nonnegative hours worked; the injury rate is per million hours.

`fatalities`
: Nonnegative integer count. Optional NULL means unknown, never zero. Subset and population consistency rules are enforced.

`fatalities_from_ill_health`
: Nonnegative integer count. Optional NULL means unknown, never zero. Subset and population consistency rules are enforced.

`recordable_ill_health_cases`
: Nonnegative integer count. Optional NULL means unknown, never zero. Subset and population consistency rules are enforced.

`days_lost`
: Nonnegative integer count. Optional NULL means unknown, never zero. Subset and population consistency rules are enforced.

`covered_workers`
: Nonnegative integer count. Optional NULL means unknown, never zero. Subset and population consistency rules are enforced.

`total_workers`
: Nonnegative integer count. Optional NULL means unknown, never zero. Subset and population consistency rules are enforced.

`audited_covered_workers`
: Nonnegative integer count. Optional NULL means unknown, never zero. Subset and population consistency rules are enforced.

`context`
: Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Accident frequency, separately classified fatalities, days lost and coverage. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Severity per million hours is analytical; fatalities and illness remain separate counts See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_ESRS_2023; ESRS S1-14 paragraph88; AR80-95. Kind: METRIC_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "injury", cases)[[1]]
args <- case$kwargs
```

```
args$context <- do.call(esg_context, case$context)
result <- do.call(health_safety_metrics, args)
stopifnot(result$status == "OK")
result$value
```

`list_resources` *List packaged resource names and checksums, without any remote registry*

Description

DATA-003: List packaged resource names and checksums, without any remote registry.

Usage

```
list_resources(context = NULL)
```

Arguments

`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

List packaged resource names and checksums, without any remote registry. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_DATA_CONTRACT_V2; CONTRACTS.md; API.md. Kind: INPUT_CONTRACT.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "resources", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(list_resources, args)
stopifnot(result$status == "OK")
result$value
```

`load_local_data` *Read UTF-8 JSON or CSV; CSV fields stay strings, with no guessed units or types*

Description

DATA-002: Read UTF-8 JSON or CSV; CSV fields stay strings, with no guessed units or types.

Usage

```
load_local_data(path, format = NULL, schema = NULL, context = NULL)
```

Arguments

`path`	Local UTF-8 JSON or CSV path. URLs are rejected.
`format`	Explicit json or csv; NULL derives it from the local extension.
`schema`	Named fields mapping with explicit type, optional required, enum, minimum, maximum and nested schema; optional unique_key and additional_fields policy.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Read UTF-8 JSON or CSV; CSV fields stay strings, with no guessed units or types. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_DATA_CONTRACT_V2; CONTRACTS.md; API.md. Kind: INPUT_CONTRACT.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
p <- tempfile(fileext = ".json")
writeLines('[{"id":"a","value":2}]', p)
load_local_data(p)
unlink(p)
```

---

load_profile — *Return an explicitly requested profile; never infer legal applicability*

Description

DATA-004: Return an explicitly requested profile; never infer legal applicability.

Usage

```
load_profile(profile_id, context = NULL)
```

Arguments

`profile_id`	Exact profile identifier from profiles.json; no automatic applicability selection.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Return an explicitly requested profile; never infer legal applicability. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_DATA_CONTRACT_V2; CONTRACTS.md; API.md. Kind: INPUT_CONTRACT.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "profile", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(load_profile, args)
stopifnot(result$status == "OK")
result$value
```

---

load_reference_data *Read and integrity-check a packaged JSON reference resource*

Description

DATA-005: Read and integrity-check a packaged JSON reference resource.

Usage

```
load_reference_data(name, context = NULL)
```

Arguments

name	Registered JSON resource basename; checked against the packaged SHA-256 manifest.
context	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Read and integrity-check a packaged JSON reference resource. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_DATA_CONTRACT_V2; CONTRACTS.md; API.md. Kind: INPUT_CONTRACT.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "reference", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(load_reference_data, args)
stopifnot(result$status == "OK")
result$value
```

---

metric_change — *Calculate change only on a declared comparable boundary/method; negative bases have no relative rate*

Description

ENV-007: Calculate change only on a declared comparable boundary/method; negative bases have no relative rate.

Usage

```
metric_change(baseline, current, comparable, unit = "value", restatement = NULL,
    context = NULL)
```

Arguments

`baseline`	Finite original baseline, signed where meaningful, or NULL.
`current`	Finite current observation, signed where meaningful, or NULL.
`comparable`	Required logical comparability decision. FALSE suppresses a misleading change.
`unit`	Explicit case-sensitive physical or observation unit. See the corresponding row or model contract.
`restatement`	Optional named baseline and reason for an explicit restatement; original baseline remains in results.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Calculate change only on a declared comparable boundary/method; negative bases have no relative rate. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_ESRS_2023; ESRS 1 section7.1 comparatives; explicit arithmetic change. Kind: ANALYTICAL_METHOD.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "change", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(metric_change, args)
stopifnot(result$status == "OK")
result$value
```

---

normalize_indicator — *Interpolate an explicit model scale; reject incompatible units and out-of-domain inputs*

Description

RATE-002: Interpolate an explicit model scale; reject incompatible units and out-of-domain inputs.

Usage

```
normalize_indicator(indicator_id, value, unit, model = NULL, context = NULL)
```

Arguments

`indicator_id`	Registered indicator identifier: integer 1..18 for PAI, or the exact rating model indicator name.
`value`	Finite scalar observation in the declared unit, or NULL for missing data.
`unit`	Explicit case-sensitive physical or observation unit. See the corresponding row or model contract.
`model`	Named model list; NULL loads rating_reference.json. Includes normative origin, entity type, positive E/S/G weights, per-pillar indicator weights, units, piecewise knots, coverage threshold, maximum age, limits and descending classes.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Interpolate an explicit model scale; reject incompatible units and out-of-domain inputs. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Normative performance model, not a statutory rating or empirically calibrated financial-risk model See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_REFERENCE_MODEL_V1; RATING.md; rating_reference.json. Kind: MODEL_CHOICE.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "normalize", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(normalize_indicator, args)
stopifnot(result$status == "OK")
result$value
```

`overall_rating` *Combine E/S/G pillar records, check coverage, then apply caps and class thresholds*

Description

RATE-005: Combine E/S/G pillar records, check coverage, then apply caps and class thresholds.

Usage

```
overall_rating(pillars, findings, model = NULL, context = NULL)
```

Arguments

`pillars`	Named E, S and G lists, each containing score (0..100 or NULL) and coverage (0..1).
`findings`	Named logical findings for rating limits, or governance finding rows with finding_id, assessment and optional topic. Missing required evidence remains unknown.
`model`	Named model list; NULL loads rating_reference.json. Includes normative origin, entity type, positive E/S/G weights, per-pillar indicator weights, units, piecewise knots, coverage threshold, maximum age, limits and descending classes.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Combine E/S/G pillar records, check coverage, then apply caps and class thresholds. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Normative performance model, not a statutory rating or empirically calibrated financial-risk model See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_REFERENCE_MODEL_V1; RATING.md; rating_reference.json. Kind: MODEL_CHOICE.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "overall", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(overall_rating, args)
stopifnot(result$status == "OK")
result$value
```

---

pay_gap — *Unadjusted pay gap (male minus female)/male, in percentage points; signed*

Description

SOC-003: Unadjusted pay gap (male minus female)/male, in percentage points; signed.

Usage

```
pay_gap(male_gross_hourly_pay, female_gross_hourly_pay, context = NULL)
```

Arguments

`male_gross_hourly_pay`
: Nonnegative gross hourly pay in the same currency and reporting boundary; NULL is missing. The gap is signed percent relative to male pay.

`female_gross_hourly_pay`
: Nonnegative gross hourly pay in the same currency and reporting boundary; NULL is missing. The gap is signed percent relative to male pay.

`context`
: Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Unadjusted pay gap (male minus female)/male, in percentage points; signed. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_ESRS_2023; ESRS S1-16 paragraph97(a); AR98. Kind: METRIC_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "paygap", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(pay_gap, args)
stopifnot(result$status == "OK")
result$value
```

---

pillar_scores — *Weight normalized indicator scores; report coverage separately and enforce minima*

Description

RATE-003: Weight normalized indicator scores; report coverage separately and enforce minima.

Usage

```
pillar_scores(scores, model = NULL, context = NULL)
```

Arguments

`scores`	Named normalized indicator scores in 0..100. NULL entries remain missing and reduce coverage.
`model`	Named model list; NULL loads rating_reference.json. Includes normative origin, entity type, positive E/S/G weights, per-pillar indicator weights, units, piecewise knots, coverage threshold, maximum age, limits and descending classes.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Weight normalized indicator scores; report coverage separately and enforce minima. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Normative performance model, not a statutory rating or empirically calibrated financial-risk model See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_REFERENCE_MODEL_V1; RATING.md; rating_reference.json. Kind: MODEL_CHOICE.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "pillars", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(pillar_scores, args)
stopifnot(result$status == "OK")
result$value
```

---

portfolio_concentration

Herfindahl index and largest group weight for nonnegative exposures

---

Description

PORT-004: Herfindahl index and largest group weight for nonnegative exposures.

Usage

```
portfolio_concentration(rows, group_field = "issuer_id", context = NULL)
```

Arguments

`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`group_field`	Explicit portfolio grouping field; defaults to issuer_id.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Herfindahl index and largest group weight for nonnegative exposures. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_METHODS_V2; CONTRACTS.md portfolio methods. Kind: ANALYTICAL_METHOD.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "concentration", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(portfolio_concentration, args)
stopifnot(result$status == "OK")
result$value
```

`portfolio_coverage` *Amount-weighted nonmissing field coverage with numerator/denominator retained*

Description

PORT-002: Amount-weighted nonmissing field coverage with numerator/denominator retained.

Usage

```
portfolio_coverage(rows, field, context = NULL)
```

Arguments

`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`field`	Explicit nonmissing-coverage or logical-exposure field in each portfolio row.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Amount-weighted nonmissing field coverage with numerator/denominator retained. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_METHODS_V2; CONTRACTS.md portfolio methods. Kind: ANALYTICAL_METHOD.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "coverage", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(portfolio_coverage, args)
stopifnot(result$status == "OK")
result$value
```

`portfolio_exposure` *Known positive exposure and unknown share; incomplete data do not become false*

Description

PORT-003: Known positive exposure and unknown share; incomplete data do not become false.

Usage

```
portfolio_exposure(rows, field, context = NULL)
```

Arguments

`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`field`	Explicit nonmissing-coverage or logical-exposure field in each portfolio row.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Known positive exposure and unknown share; incomplete data do not become false. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_METHODS_V2; CONTRACTS.md portfolio methods. Kind: ANALYTICAL_METHOD.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "exposure", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(portfolio_exposure, args)
stopifnot(result$status == "OK")
result$value
```

---

portfolio_rating_summary

Value-weight compatible company scores; report classes and unrated exposure separately

---

Description

RATE-009: Value-weight compatible company scores; report classes and unrated exposure separately.

Usage

```
portfolio_rating_summary(rows, context = NULL)
```

Arguments

`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Value-weight compatible company scores; report classes and unrated exposure separately. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Normative performance model, not a statutory rating or empirically calibrated financial-risk model See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_REFERENCE_MODEL_V1; RATING.md; rating_reference.json. Kind: MODEL_CHOICE.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
c <- Filter(function(x) x$id == "company", cases)[[1]]
ctx <- do.call(esg_context, c$context)
r <- do.call(rate_company, c(c$kwargs, list(context = ctx)))
rows <- list(list(position_id = "p", issuer_id = "i", value = 100,
  currency = ctx$currency, as_of = ctx$as_of, rating = r))
portfolio_rating_summary(rows, context = ctx)
```

`position_weights` *Long-only weights over the entire declared portfolio; no missing-position deletion*

Description

PORT-001: Long-only weights over the entire declared portfolio; no missing-position deletion.

Usage

```
position_weights(rows, context = NULL)
```

Arguments

`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Long-only weights over the entire declared portfolio; no missing-position deletion. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_METHODS_V2; CONTRACTS.md portfolio methods. Kind: ANALYTICAL_METHOD.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "weights", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(position_weights, args)
stopifnot(result$status == "OK")
result$value
```

---

print.esg_result *Print a structured calculation result*

Description

Display function ID, execution status, unit and value.

Usage

```
## S3 method for class 'esg_result'
print(x, ...)
```

Arguments

x	An esg_result returned by a calculation.
...	Reserved for generic compatibility.

Value

The unchanged result, invisibly.

Examples

```
print(pay_gap(20, 18))
```

---

rate_company — *Rate dated sourced observations with coverage, provenance, contributions and binding caps*

Description

RATE-006: Rate dated sourced observations with coverage, provenance, contributions and binding caps.

Usage

```
rate_company(observations, findings, entity_type = "nonfinancial",
    model = NULL, context = NULL)
```

Arguments

observations	Named dated rating observations with value, unit, as_of and source, or PAI observation rows with as_of, indicator_id, component, unit, method_id, status and value.
findings	Named logical findings for rating limits, or governance finding rows with finding_id, assessment and optional topic. Missing required evidence remains unknown.
entity_type	Declared model entity type, default nonfinancial.
model	Named model list; NULL loads rating_reference.json. Includes normative origin, entity type, positive E/S/G weights, per-pillar indicator weights, units, piecewise knots, coverage threshold, maximum age, limits and descending classes.
context	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Rate dated sourced observations with coverage, provenance, contributions and binding caps. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Normative performance model, not a statutory rating or empirically calibrated financial-risk model See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_REFERENCE_MODEL_V1; RATING.md; rating_reference.json. Kind: MODEL_CHOICE.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "company", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(rate_company, args)
stopifnot(result$status == "OK")
result$value
```

`rating_sensitivity` *Recalculate explicit indicator-weight scenarios; preserve original model and observations*

Description

RATE-008: Recalculate explicit indicator-weight scenarios; preserve original model and observations.

Usage

```
rating_sensitivity(observations, findings, scenarios, model = NULL, context = NULL)
```

Arguments

`observations`	Named dated rating observations with value, unit, as_of and source, or PAI observation rows with as_of, indicator_id, component, unit, method_id, status and value.
`findings`	Named logical findings for rating limits, or governance finding rows with finding_id, assessment and optional topic. Missing required evidence remains unknown.
`scenarios`	Rows with scenario_id, reason and optional indicator_weights, pillar_weights, observations or findings overrides. Original inputs are not mutated.
`model`	Named model list; NULL loads rating_reference.json. Includes normative origin, entity type, positive E/S/G weights, per-pillar indicator weights, units, piecewise knots, coverage threshold, maximum age, limits and descending classes.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Recalculate explicit indicator-weight scenarios; preserve original model and observations. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Normative performance model, not a statutory rating or empirically calibrated financial-risk model See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_REFERENCE_MODEL_V1; RATING.md; rating_reference.json. Kind: MODEL_CHOICE.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
c <- Filter(function(x) x$id == "company", cases)[[1]]
a <- c$kwargs
a$context <- do.call(esg_context, c$context)
a$scenarios <- list(list(scenario_id = "same", reason = "Explicit unchanged scenario"))
rating <- do.call(rating_sensitivity, a)
stopifnot(rating$value[[1]]$delta == 0)
```

reconcile_kpi_allocations

Reconcile disjoint allocation fractions against original amount IDs

Description

KPI-004: Reconcile disjoint allocation fractions against original amount IDs.

Usage

```
reconcile_kpi_allocations(amounts, allocations, context = NULL)
```

Arguments

`amounts`	Rows containing unique amount_id and nonnegative amount in context currency.
`allocations`	Disjoint rows with unique allocation_id, amount_id and fraction in 0..1; taxonomy contributions additionally require distinct goal codes.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Reconcile disjoint allocation fractions against original amount IDs. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_TAX_KPI_CONSOLIDATED_20260101; Annex I 1.2.2; Annex II: avoid double counting. Kind: LEGAL_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "allocation_reconcile", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(reconcile_kpi_allocations, args)
stopifnot(result$status == "OK")
result$value
```

`taxonomy_capex` *Compute CapEx KPI with accounting exclusions, plan and individual-measure routes*

Description

KPI-002: Compute CapEx KPI with accounting exclusions, plan and individual-measure routes.

Usage

```
taxonomy_capex(rows, prior_periods = list(), context = NULL)
```

Arguments

`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`prior_periods`	Historical period records with period_end, optional period_start, rows and previously_reported_aligned_numerator; historical plans require a current plan_id review.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Compute CapEx KPI with accounting exclusions, plan and individual-measure routes. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_TAX_KPI_CONSOLIDATED_20260101; Article 2(1b); Annex I 1.1.2.1-1.1.2.2; 2026/73 Article 4 election. Kind: LEGAL_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "capex", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(taxonomy_capex, args)
stopifnot(result$status == "OK")
result$value
```

---

taxonomy_opex — *Compute OpEx KPI, excluding capitalised/double-counted expenditure*

Description

KPI-003: Compute OpEx KPI, excluding capitalised/double-counted expenditure.

Usage

```
taxonomy_opex(rows, immaterial_reason = NULL, context = NULL)
```

Arguments

`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`immaterial_reason`	Explicit nonempty explanation for the OpEx immateriality exemption, or NULL for normal calculation.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Compute OpEx KPI, excluding capitalised/double-counted expenditure. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_TAX_KPI_CONSOLIDATED_20260101; Article 2(1c); Annex I 1.1.3.1-1.1.3.2. Kind: LEGAL_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "opex", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(taxonomy_opex, args)
stopifnot(result$status == "OK")
result$value
```

---

taxonomy_turnover — *Compute nonfinancial turnover KPI from net-revenue rows and explicit assessments*

Description

KPI-001: Compute nonfinancial turnover KPI from net-revenue rows and explicit assessments.

Usage

```
taxonomy_turnover(rows, context = NULL)
```

Arguments

`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Compute nonfinancial turnover KPI from net-revenue rows and explicit assessments. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_TAX_KPI_CONSOLIDATED_20260101; Article 2(1a); Article 8(6)-(7); Annex I 1.1.1. Kind: LEGAL_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "turnover", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(taxonomy_turnover, args)
stopifnot(result$status == "OK")
result$value
```

`validate_activity_assessment` *Validate distinct activities and their assessment inputs without classifying NACE codes*

Description

TAX-003: Validate distinct activities and their assessment inputs without classifying NACE codes.

Usage

```
validate_activity_assessment(activities, context = NULL)
```

Arguments

`activities`	Rows with activity_id, optional period_end, eligible, sc, dnsh, minimum_safeguards and technical_screening assessments.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Validate distinct activities and their assessment inputs without classifying NACE codes. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_TAX_BASE_2020; Articles 3, 9, 17, 18. Kind: LEGAL_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "activity", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(validate_activity_assessment, args)
stopifnot(result$status == "OK")
result$value
```

---

validate_governance_findings

Check provenance/status of structured findings, without investigating their truth

---

Description

GOV-002: Check provenance/status of structured findings, without investigating their truth.

Usage

```
validate_governance_findings(findings, context = NULL)
```

Arguments

`findings`	Named logical findings for rating limits, or governance finding rows with finding_id, assessment and optional topic. Missing required evidence remains unknown.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Check provenance/status of structured findings, without investigating their truth. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

No independent investigation or legal determination See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_ESRS_2023; ESRS G1 structured supplied findings. Kind: INPUT_CONTRACT.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "findings", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(validate_governance_findings, args)
stopifnot(result$status == "OK")
result$value
```

---

validate_inputs — *Validate explicit nested table schemas, types, enums, units, bounds and unique IDs*

Description

DATA-001: Validate explicit nested table schemas, types, enums, units, bounds and unique IDs.

Usage

```
validate_inputs(rows, schema, context = NULL)
```

Arguments

`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`schema`	Named fields mapping with explicit type, optional required, enum, minimum, maximum and nested schema; optional unique_key and additional_fields policy.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Validate explicit nested table schemas, types, enums, units, bounds and unique IDs. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_DATA_CONTRACT_V2; CONTRACTS.md; API.md. Kind: INPUT_CONTRACT.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "schema", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(validate_inputs, args)
stopifnot(result$status == "OK")
result$value
```

---

validate_pai_inputs — *Check an indicator's declared universe, consistent issuer data and required typed fields*

Description

PAI-004: Check an indicator's declared universe, consistent issuer data and required typed fields.

Usage

```
validate_pai_inputs(indicator_id, rows, context = NULL)
```

Arguments

`indicator_id`	Registered indicator identifier: integer 1..18 for PAI, or the exact rating model indicator name.
`rows`	List of named row lists, or a data.frame with explicit scalar column types. Required IDs must be unique. The complete row schema is in the installed contracts document and the executable example.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Check an indicator's declared universe, consistent issuer data and required typed fields. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Weighted means for 5/6/12/13 are declared method choices; PAI18 PED ambiguity has explicit interpretations See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_SFDR_RTS_CONSOLIDATED_20230220; Annex I Table 1 indicators 1-18 and definitions; ESA JC2023_18 Section IV questions 10,11,17,27,28. Kind: LEGAL_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "pai_validation", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(validate_pai_inputs, args)
stopifnot(result$status == "OK")
result$value
```

---

validate_rating_model *Validate bounded piecewise scales, complete pillars, weights, classes and caps*

Description

RATE-001: Validate bounded piecewise scales, complete pillars, weights, classes and caps.

Usage

```
validate_rating_model(model = NULL, context = NULL)
```

Arguments

`model`	Named model list; NULL loads rating_reference.json. Includes normative origin, entity type, positive E/S/G weights, per-pillar indicator weights, units, piecewise knots, coverage threshold, maximum age, limits and descending classes.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Validate bounded piecewise scales, complete pillars, weights, classes and caps. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Normative performance model, not a statutory rating or empirically calibrated financial-risk model See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

ESGCALC_REFERENCE_MODEL_V1; RATING.md; rating_reference.json. Kind: MODEL_CHOICE.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "rating_model", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(validate_rating_model, args)
stopifnot(result$status == "OK")
result$value
```

`waste_resource_metrics`

Separate waste destinations/hazardous totals and secondary material input shares

Description

ENV-006: Separate waste destinations/hazardous totals and secondary material input shares.

Usage

```
waste_resource_metrics(waste_rows, material_inputs = list(), context = NULL)
```

Arguments

`waste_rows`	Rows with waste_id, value, unit, hazardous and destination (reused, recycled, other_recovery or disposal).
`material_inputs`	Rows with material_id, value, unit, secondary and optional reused flags; reused input must be secondary.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Separate waste destinations/hazardous totals and secondary material input shares. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Structured scope/applicability supplied by the caller See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_ESRS_2023; ESRS E5-4 paragraph31; E5-5 paragraphs35-40. Kind: METRIC_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "waste", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(waste_resource_metrics, args)
stopifnot(result$status == "OK")
result$value
```

`water_metrics` *Common-boundary consumption, risk-area consumption, storage and revenue intensity*

Description

ENV-005: Common-boundary consumption, risk-area consumption, storage and revenue intensity.

Usage

```
water_metrics(withdrawn = NULL, discharged = NULL, reused = NULL,
    unit = "m3", facilities = NULL, stored_start = NULL, stored_end = NULL,
    net_revenue_eur = NULL, context = NULL)
```

Arguments

`withdrawn`	Nonnegative water quantity in unit, or NULL. Storage observations must be supplied together; facility and aggregate quantities cannot be mixed.
`discharged`	Nonnegative water quantity in unit, or NULL. Storage observations must be supplied together; facility and aggregate quantities cannot be mixed.
`reused`	Nonnegative water quantity in unit, or NULL. Storage observations must be supplied together; facility and aggregate quantities cannot be mixed.
`unit`	Explicit case-sensitive physical or observation unit. See the corresponding row or model contract.
`facilities`	Facility rows with facility_id, withdrawn, discharged, reused, unit and water_risk; mutually exclusive with aggregate water quantities.
`stored_start`	Nonnegative water quantity in unit, or NULL. Storage observations must be supplied together; facility and aggregate quantities cannot be mixed.
`stored_end`	Nonnegative water quantity in unit, or NULL. Storage observations must be supplied together; facility and aggregate quantities cannot be mixed.
`net_revenue_eur`	Nonnegative net revenue in EUR, used for water intensity per million EUR.
`context`	Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Common-boundary consumption, risk-area consumption, storage and revenue intensity. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Reuse/withdrawal-plus-reuse share is a separate analytical convention See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_ESRS_2023; ESRS E3-4 paragraphs 28-29; AR28-32. Kind: METRIC_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "water", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(water_metrics, args)
stopifnot(result$status == "OK")
result$value
```

---

workforce_metrics *Aggregate disjoint workforce groups in headcount and FTE separately*

Description

SOC-001: Aggregate disjoint workforce groups in headcount and FTE separately.

Usage

```
workforce_metrics(groups, leavers = NULL, turnover_denominator = NULL,
    denominator_method = NULL, context = NULL)
```

Arguments

- `groups` — Disjoint workforce rows (group_id, headcount, fte) or diversity rows (category, count). No inferred population overlap.
- `leavers` — Nonnegative integer count. Optional NULL means unknown, never zero. Subset and population consistency rules are enforced.
- `turnover_denominator` — Nonnegative explicitly selected employee denominator for leavers.
- `denominator_method` — Nonempty declared method for the turnover population denominator.
- `context` — Calculation context from esg_context(); NULL uses the undated analytical methods profile. Legal profiles require explicit dates.

Details

Aggregate disjoint workforce groups in headcount and FTE separately. Numeric inputs must be finite scalars; factors and implicit vector recycling are rejected. NULL or scalar NA is missing, not zero. Invalid contracts raise classified `esg_error` conditions with a `code` and optional `field`.

Turnover denominator methodology explicitly selected and disclosed See esgcalc-contracts for row schemas, algorithms, units and boundary rules. The executable example uses a fixed independent reference case; inspect `args` to see all nested fields. External evidence is structurally validated, not independently investigated.

Value

An `esg_result` named list. Its `value` and `unit` carry the result; `status` distinguishes OK, missing, partial, not applicable, unrated and excluded outcomes. Source IDs, profile, mode, input hash and detailed denominators or contributions are retained. See esgcalc-contracts.

References

EU_ESRS_2023; ESRS S1-6 paragraphs50-52; AR59. Kind: METRIC_DEFINITION.

See Also

esg_context, load_reference_data, esgcalc-contracts

Examples

```
cases <- load_reference_data("reference_cases.json")$value$cases
case <- Filter(function(x) x$id == "workforce", cases)[[1]]
args <- case$kwargs
args$context <- do.call(esg_context, case$context)
result <- do.call(workforce_metrics, args)
stopifnot(result$status == "OK")
result$value
```

Index

∗ **package**
esgcalc-package, 3

aggregate_ghg_inventory, 4
aggregate_issuer_positions, 5
aggregate_pai_observations, 6
aggregate_pai_portfolio, 7
allocate_multi_objective_contributions, 9
apply_rating_limits, 10

board_metrics, 11

calculate_pai_indicator, *3*, 13
check_quantitative_criterion, *3*, 14
combine_taxonomy_assessments, 15
convert_currency, 17

diversity_metrics, 18

emissions_from_activity, 19
energy_metrics, 20
esg_context, *3*, *5–9*, *11, 12*, *14–16*, *18–21*, 22, *29, 30*, *32–35*, *37–41*, *43–47*, *49–51*, *53–55*, *57–60*, *62–66*, *68, 69*
esgcalc *(*esgcalc-package*)*, 3
esgcalc-contracts, *3–21*, 23, *29–51*, *53–66*, *68, 69*
esgcalc-package, 3
esgcalc-resources, *3*, 27
explain_rating, 29
export_resources, *28*, 30

financed_emissions, 31

ghg_intensity, 32

health_safety_metrics, 33

list_resources, *3*, *28*, 35
load_local_data, 36
load_profile, 37
load_reference_data, *5–9*, *11, 12*, *14–16*, *18–21*, *28–30*, *32–35*, *37*, 38, *38–41*, *43–47*, *49–51*, *53–55*, *57–60*, *62–66*, *68, 69*

metric_change, 39

normalize_indicator, 40

overall_rating, 42

pay_gap, *3*, 43
pillar_scores, 44
portfolio_concentration, 45
portfolio_coverage, 47
portfolio_exposure, 48
portfolio_rating_summary, 49
position_weights, 50
print.esg_result, 51

rate_company, *3*, 52
rating_sensitivity, 53
reconcile_kpi_allocations, 55

taxonomy_capex, 56
taxonomy_opex, 57
taxonomy_turnover, 58

validate_activity_assessment, 60
validate_governance_findings, 61
validate_inputs, 62
validate_pai_inputs, 63
validate_rating_model, 64

waste_resource_metrics, 66
water_metrics, 67
workforce_metrics, 68