Package 'tikatuwq'

October 18, 2026

Type Package

Title Water Quality Assessment and Environmental Compliance in Brazil

Version 0.11.0

Maintainer Vinicius Saraiva Santos <vinisaraiva@gmail.com>

Description Tools to import, clean, validate, and analyze freshwater quality data in Brazil. Implements water quality indices including the Water Quality Index ('WQI'/'IQA') using the weighted geometric mean following 'CETESB' methodology, an independent engine reproducing the 'IQA' of the Minas Gerais water agency 'IGAM' (a documented reproduction, not endorsed by the agency), paired comparison of 'IQA' engines with compare_iqa(), the Trophic State Index ('TSI'/'IET') after Carlson (1977) <doi:10.4319/lo.1977.22.2.0361> and Lamparelli (2004) <https://teses.usp.br/teses/disponiveis/41/41134/tde-20032006-075813/publico/TeseLamparelli2004.pdf>, and the National Sanitation Foundation Water Quality Index ('NSF WQI', Brown (1970)). The package also checks compliance with Brazilian standard 'CONAMA' Resolution 357/2005 <https://conama.mma.gov.br/?id=450&option=com_sisconama&task=arquivo.download> including the legal frequency rule (Art. 15, 80% conformity over six or more samples per year), and provides seasonal analysis with regional flow-season calendars, pollutant load computation, exceedance probability estimation, 'IET' visualization, and multivariate 'PCA' tools for routine monitoring workflows. The example dataset ('wq_demo') is a real subset from 'INEMA' monitoring data from a river in Bahia, Brazil (2020-2024).

License MIT + file LICENSE

Encoding UTF-8

Language en-US

LazyData true

LazyDataCompression xz

Depends R (>= 4.1)

Imports dplyr, readr, tibble, rlang, stats, utils, ggplot2, tidyr, lubridate, stringr, glue, scales, broom, purrr, tools

Suggests testthat (>= 3.0.0), spelling, rmarkdown, knitr, pkgdown, leaflet, withr

VignetteBuilder knitr

URL https://github.com/tikatuwq/tikatuwq, https://tikatuwq.github.io/tikatuwq/

BugReports https://github.com/tikatuwq/tikatuwq/issues

Config/testthat/edition 3

RoxygenNote 7.3.3

NeedsCompilation no

Author Vinicius Saraiva Santos [aut, cre] (ORCID: <https://orcid.org/0009-0007-1387-7927>), Fabricio Berton Zanchi [ctb] (ORCID: <https://orcid.org/0000-0002-8074-2893>)

Repository CRAN

Date/Publication 2026-10-17 22:40:02 UTC

Contents

assign_season . . . 3
balnear_check . . . 5
classify_iqa . . . 7
classify_tsi_carlson . . . 8
classify_tsi_lamparelli . . . 9
clean_units . . . 9
compare_iqa . . . 10
compare_seasons . . . 12
compute_load . . . 13
conama_check . . . 15
conama_freq_check . . . 16
conama_limits . . . 17
conama_report . . . 18
conama_summary . . . 19
conama_text . . . 20
exceedance_prob . . . 21
fix_coords . . . 22
generate_analysis . . . 23
iet_carlson . . . 24
iet_lamparelli . . . 26
iqa . . . 27
iqa_components . . . 31
iqa_methods . . . 33
mk_seasonal . . . 34
normalize_param_names . . . 36
nsfwqi . . . 36

param_plot . . . 38
param_plot_multi . . . 39
param_summary . . . 40
param_summary_multi . . . 41
param_trend . . . 42
param_trend_multi . . . 43
plot_box . . . 44
plot_heatmap . . . 44
plot_iet . . . 45
plot_iqa . . . 46
plot_map . . . 47
plot_map_quality . . . 48
plot_series . . . 50
plot_trend . . . 50
read_wq . . . 52
render_report . . . 53
resume_wq . . . 55
trend_param . . . 56
validate_wq . . . 57
wq_demo . . . 58
wq_pca . . . 59

Index **61**

`assign_season` *Atribui periodo hidrologico (chuvoso / seco) a cada amostra*

Description

Adiciona a coluna `season` ao data frame, classificando cada amostra como `"chuvoso"` ou `"seco"` com base no mes de coleta e no calendario hidrologico regional brasileiro.

Usage

```
assign_season(
  df,
  region = NULL,
  date_col = "data",
  wet_months = NULL,
  labels = c("chuvoso", "seco")
)
```

Arguments

`df`	Data frame com ao menos a coluna `date_col`.

region	Character ou NULL (default); regiao climatica usada para definir o calendario chuvoso/seco. Uma de "sudeste", "bahia", "centro_oeste", "nordeste", "norte", "sul" ou "custom". Este argumento e **opcional**: muitas analises de qualidade da agua nao consideram sazonalidade regional. Se region = NULL: • se wet_months for fornecido, ele e usado diretamente (equivalente a region = "custom"); • caso contrario, a coluna season e preenchida com NA e uma mensagem informativa e exibida (sem erro), de forma que o restante do fluxo de analise nao e bloqueado.
date_col	Character; nome da coluna de datas. Default "data".
wet_months	Integer vector; meses numericos do periodo chuvoso (1 = Jan ... 12 = Dez). Obrigatorio quando region = "custom". Tambem pode ser usado com region = NULL para definir um calendario personalizado sem escolher uma regiao pre-definida.
labels	Character vector de comprimento 2 com os rotulos para os periodos chuvoso e seco, nesta ordem. Default c("chuvoso", "seco").

Details

Os calendarios regionais disponiveis (meses chuvosos) sao:

"sudeste" Outubro-Marco (Oct-Mar).

"bahia" Outubro-Marco; adequado para BA litoral sul, regiao do Rio Buranhem, ES.

"centro_oeste" Outubro-Marco.

"nordeste" Fevereiro-Julho; semiarido/intertropical.

"norte" Dezembro-Maio; Amazonia.

"sul" Junho-Setembro; inverno umido subtropical.

"custom" Define os meses chuvosos pelo argumento wet_months.

Value

O df de entrada com a coluna season adicionada (character, podendo ser NA quando nenhuma regiao/calendario for informado).

See Also

compare_seasons()

Other seasonal-tools: compare_seasons()

Examples

```
data("wq_demo", package = "tikatuwq")
d <- assign_season(wq_demo, region = "bahia")
table(d$season)

# region e opcional: sem informa-la, season fica NA (sem erro)
d2 <- assign_season(wq_demo)
```

```
table(d2$season, useNA = "always")
```

`balnear_check` *Avaliacao de balneabilidade — CONAMA 274/2000*

Description

Classifica pontos de amostragem quanto a balneabilidade para recreacao de contato primario (banho, natacao, mergulho), conforme a Resolucao CONAMA 274/2000. A classificacao e baseada na regra dos 80%: um ponto e considerado proprio em determinada categoria quando pelo menos 80% das ultimas `n_samples` amostras estao dentro do limite correspondente.

Usage

```
balnear_check(
  df,
  col = NULL,
  by = "ponto",
  date_col = "data",
  n_samples = 5L,
  threshold_pct = 0.8,
  locale = c("pt", "en")
)
```

Arguments

`df`	Data frame com ao menos a coluna do indicador microbiologico e as colunas de agrupamento.
`col`	Character; nome da coluna do indicador. Se `NULL` (default), detectada automaticamente a partir de nomes comuns: `coliformes`, `col_termotolerantes`, `e_coli`, `ecoli`.
`by`	Character vector; colunas de agrupamento. Default `"ponto"`.
`date_col`	Character; nome da coluna de datas, usada para selecionar as `n_samples` amostras mais recentes. Default `"data"`.
`n_samples`	Integer; numero de amostras mais recentes a considerar. Default `5L` (referencia CONAMA 274/2000).
`threshold_pct`	Numeric; fracao minima de conformidade para classificar como "propria". Default `0.80` (80%).
`locale`	Character; idioma dos rotulos: `"pt"` (default) ou `"en"`.

Details

Categorias (aguas doces, contato primario):

Excelente Colif. termotolerantes $\leq$ 250 NMP/100mL (ou E. coli $\leq$ 200) em $\geq$ 80% das amostras.

Muito Boa Colif. $\leq$ 500 (ou E. coli $\leq$ 400) em $\geq$ 80%.

Satisfatoria Colif. $\leq$ 1000 (ou E. coli $\leq$ 800) em $\geq$ 80%.

Impropria Mais de 20% das amostras ultrapassam o limite de "Satisfatoria".

A avaliacao usa as `n_samples` amostras mais recentes por grupo. Se o grupo tiver menos amostras, a classificacao e realizada mesmo assim mas a coluna `amostras_insuficientes` sera `TRUE`. Recomenda-se ao menos 5 amostras (referencia: "ultimas 5 semanas" do texto da norma).

Value

Um tibble com uma linha por grupo, contendo:

indicador Coluna usada para a avaliacao.

n_amostras Total de amostras disponiveis no grupo.

n_avaliadas Amostras usadas na avaliacao (`min(n_amostras, n_samples)`).

pct_ok_satisfatoria Fracao dentro do limite "Satisfatoria".

classificacao Uma de: "Excelente", "Muito Boa", "Satisfatoria", "Impropria".

propria Logical; `TRUE` para as tres primeiras categorias.

amostras_insuficientes Logical; `TRUE` quando `n_amostras < n_samples`.

References

CONAMA (2000). Resolucao 274, de 29 de novembro de 2000. Ministerio do Meio Ambiente, Brasilia. Diario Oficial da Uniao 18/01/2001.

See Also

`conama_check()`, `conama_freq_check()`

Other conama-tools: `conama_freq_check()`

Examples

```
data("wq_demo", package = "tikatuwq")
balnear_check(wq_demo, by = "ponto")
```

`classify_iqa` *Classifica valores do IQA/WQI em faixas qualitativas*

Description

Converte valores numericos de IQA (0-100) em classes qualitativas conforme a escala institucional escolhida. A classificacao nao altera o valor numerico: ela apenas aplica cortes e rotulos a um IQA ja calculado.

Escalas disponiveis (intervalos fechados a direita, ex.: Ruim = (19, 36]):

`"cetesb"` CETESB/SP (padrao): Pessima [0, 19], Ruim (19, 36], Regular (36, 51], Boa (51, 79], Otima (79, 100] (Apendice D CETESB 2023, Tabela 1, p. 5).

`"ana"` Portal de Qualidade das Aguas da ANA (qualidadedaagua.ana.gov.br/iqa.html, consultado em 2026-10-15), que apresenta os mesmos cortes 19/36/51/79 da CETESB. E, portanto, um alias de `"cetesb"`. Faixas historicas de outros estados (ex.: 20/37/52/79) citadas em publicacoes antigas da ANA nao estao implementadas.

`"igam"` IGAM/MG: Muito Ruim [0, 25], Ruim (25, 50], Medio (50, 70], Bom (70, 90], Excelente (90, 100] (IGAM, relatorio de monitoramento da sub-bacia do ribeirao Pampulha, 3o trimestre de 2017, p. 9).

`"legacy"` Escala aproximada mantida por compatibilidade com versoes antigas do pacote (cortes 25/50/70/90, rotulos Muito ruim/Ruim/ Regular/Boa/Otima). Nao corresponde a nenhuma escala institucional verificada.

Valores `NA`/`NaN` resultam em `NA`. Valores fora de [0, 100] (inclusive `Inf`) tambem resultam em `NA`, com aviso. A classificacao e feita sobre o valor recebido, sem arredondamento adicional.

Usage

```
classify_iqa(
  x,
  classification = c("cetesb", "igam", "ana", "legacy"),
  locale = c("pt", "en"),
  scheme = NULL
)
```

Arguments

`x`	Vetor numerico com IQA em 0-100.
`classification`	Escala de classificacao: `"cetesb"` (padrao), `"igam"`, `"ana"` ou `"legacy"`.
`locale`	Idioma dos rotulos: `"pt"` (padrao) ou `"en"`.
`scheme`	Depreciado (desde 0.11.0); use `classification`. Se ambos forem informados com valores diferentes, ocorre erro.

Value

Um fator ordenado com os rotulos de classe.

Note

Usar `classification = "igam"` sobre um IQA calculado pelo metodo CETESB nao equivale ao IQA calculado integralmente pelo IGAM, que usa parametros (nitrato, fosfato total), qi de temperatura (92) e regra de dados ausentes proprios.

Examples

```
vals <- c(15, 28, 45, 65, 85, 95)

# Classificacao CETESB (padrao)
classify_iqa(vals)

# Classificacao IGAM (Minas Gerais)
classify_iqa(vals, classification = "igam")

# Comparar escalas lado a lado
data.frame(
  IQA    = vals,
  CETESB = classify_iqa(vals, classification = "cetesb"),
  IGAM   = classify_iqa(vals, classification = "igam")
)
```

---

classify_tsi_carlson *Classifica TSI (Carlson) em faixas qualitativas*

Description

Converte valores do indice trofico de Carlson (TSI/IET) para classes qualitativas ordenadas. Retorna fator ordenado em portugues ("pt") ou ingles ("en").

Usage

```
classify_tsi_carlson(x, locale = c("pt", "en"))
```

Arguments

`x`	Vetor numerico com TSI/IET (0-100). NA preservado.
`locale`	Idioma dos rotulos: `"pt"` (padrao) ou `"en"`.

Value

Fator ordenado com classes de trofia.

Examples

```
classify_tsi_carlson(c(25, 35, 45, 60, 80))
classify_tsi_carlson(c(25, 35, 45, 60, 80), locale = "en")
```

`classify_tsi_lamparelli` *Classifica TSI (Lamparelli) em faixas qualitativas*

Description

Converte valores do indice trofico de Lamparelli (TSI/IET) para classes qualitativas ordenadas. Retorna fator ordenado em portugues ("pt") ou ingles ("en").

Usage

```
classify_tsi_lamparelli(x, locale = c("pt", "en"))
```

Arguments

`x`	Vetor numerico com TSI/IET (0-100). NA preservado.
`locale`	Idioma dos rotulos: `"pt"` (padrao) ou `"en"`.

Value

Fator ordenado com classes de trofia (Lamparelli).

Examples

```
classify_tsi_lamparelli(c(40, 50, 56, 61, 65, 72))
classify_tsi_lamparelli(c(40, 50, 56, 61, 65, 72), locale = "en")
```

`clean_units` *Normalize/standardize units*

Description

Normalizes units for water quality parameters. Currently handles common conversions (mg/L to µg/L for phosphorus, unit standardization). Also validates expected unit ranges and emits warnings for values outside typical ranges.

Usage

```
clean_units(df, units_map = NULL)
```

Arguments

`df`	Input data frame / tibble.
`units_map`	Optional named list mapping parameter names to target units (currently used for validation only).

Details

This function is designed as an extension point. Future versions may implement actual unit conversions based on metadata or user specifications.

Value

The input `df` with normalized units. Currently performs:

- Validation of unit ranges (warns if values are outside typical ranges)
- No actual conversions are performed (returns input unchanged)

See Also

`read_wq()`

Examples

```
df <- data.frame(ph = c(7, 7.2), od = c(6.5, 7.0), p_total = c(0.05, 0.08))
clean_units(df)
```

`compare_iqa` *Compara motores de calculo do IQA em amostras pareadas*

Description

Calcula o IQA da mesma tabela com cada motor independente (`method` de `iqa`), mantendo a identidade de cada linha, e resume as diferencas **apenas nas linhas validas para todos os metodos** (amostras pareadas).

Usage

```
compare_iqa(
  df,
  methods = c("CETESB", "IGAM"),
  id_cols = NULL,
  details = FALSE,
  missing_policy = c("paired"),
  ...
)
```

Arguments

`df`	Data frame com as observacoes (nao e modificado).
`methods`	Vetor com dois ou mais metodos distintos entre `"CETESB"` e `"IGAM"`. O primeiro e a referencia das diferencas.

id_cols	Nomes de colunas que identificam cada linha (ex.: c("ponto", "data")). Devem ser unicas em conjunto. Se NULL, usa o numero da linha (.row).
details	Logico; anexa os componentes de cada metodo.
missing_policy	Politica para linhas incompletas. Apenas "paired" (estatisticas so com linhas validas em todos os metodos).
...	Argumentos repassados a iqa para todos os metodos (ex.: altitude_m, censor_policy). allow_partial, classification, method, details e pesos nao sao aceitos.

Details

- Cada metodo usa a sua propria regra oficial de completude (allow_partial = FALSE); linhas que um metodo nao calcula ficam inelegiveis, com o motivo em motivo_<metodo>.
- Os metodos usam entradas diferentes: o CETESB usa nitrogenio total e fosforo total; o IGAM usa nitrato (com nitrate_unit), fosfato total ou fosforo total com phosphorus_basis explicito, e cloreto. Informe esses argumentos em ...; eles valem para ambos os metodos (o CETESB ignora nitrate_unit).
- Metodo indisponivel em iqa_methods nao gera erro: todas as linhas ficam inelegiveis para ele, com o motivo.
- As escalas de classe CETESB e IGAM tem rotulos e cortes diferentes. A funcao nao calcula "percentual de concordancia" comparando rotulos; em vez disso devolve a tabela de transicoes de classe (atributo "class_transitions") e as diferencas numericas.
- As diferencas usam o IQA nao arredondado. Correlacoes so sao calculadas com pelo menos 3 pares; abaixo disso ficam NA.
- details = TRUE anexa os componentes (qi, pesos efetivos) de cada metodo no atributo "components". A funcao nao atribui causa as diferencas.
- ANA nao e incluida (nao ha motor ANA distinto; ver system.file("audit", "ANA_VS_CETESB.md", package = "tikatuwq")), nem as rotas legadas/NSF, que sao aproximacoes nao validadas.

Value

Data frame com uma linha por observacao: identificadores, IQA_<metodo>, classe_<metodo>, elegivel_<metodo>, motivo_<metodo>, par_valido e, para cada metodo alem do primeiro, delta_<metodo>_minus_<ref e abs_delta_<metodo>_minus_<referencia>. Atributos: "summary" (contagens e estatisticas pareadas), "class_transitions", "methods", "missing_policy" e, se pedido, "components".

See Also

iqa, iqa_methods, classify_iqa

Examples

```
# Dados ilustrativos (nao sao medicoes reais) com as entradas dos dois
# metodos; P4 nao tem cloreto e fica inelegivel para o IGAM.
d <- data.frame(
  ponto = c("P1", "P2", "P3", "P4"),
  od = c(7, 5, 3, 6), temperatura = 24, cloreto = c(10, 30, 60, NA),
  coliformes = c(100, 2000, 20000, 500), ph = c(7.2, 7.5, 7, 7.4),
```

```
  dbo = c(2, 5, 15, 3), nt_total = c(1, 3, 8, 2), nitrato = c(0.5, 1, 2, 0.8),
  p_total = c(0.05, 0.2, 0.8, 0.1), turbidez = c(5, 20, 60, 10),
  solidos_totais = c(100, 200, 400, 150)
)
cmp <- compare_iqa(d, id_cols = "ponto", phosphorus_basis = "P",
                   nitrate_unit = "N", altitude_m = 800)
cmp[, c("ponto", "IQA_CETESB", "IQA_IGAM", "par_valido", "motivo_IGAM")]
attr(cmp, "summary")
attr(cmp, "class_transitions")
```

`compare_seasons` *Comparacao estatistica entre periodos hidrologicos*

Description

Compara um parametro de qualidade da agua entre os periodos chuvoso e seco, com estatisticas descritivas, teste de hipotese e grafico. Requer que o data frame ja tenha a coluna `season` (use `assign_season()` antes).

Usage

```
compare_seasons(
  df,
  param,
  season_col = "season",
  by = "ponto",
  test = c("wilcoxon", "t_test", "kruskal"),
  alpha = 0.05,
  plot = TRUE
)
```

Arguments

`df`	Data frame com ao menos as colunas `parametro`, `season_col` e as colunas em `by`.
`param`	Character; nome da coluna do parametro a comparar.
`season_col`	Character; nome da coluna de periodo hidrologico. Default `"season"`.
`by`	Character vector; colunas de agrupamento (ex.: `"ponto"`). Se `NULL`, analisa o conjunto todo sem agrupamento.
`test`	Metodo de comparacao: `"wilcoxon"` (default), `"t_test"` ou `"kruskal"`.
`alpha`	Nivel de significancia para classificar tendencia. Default `0.05`.
`plot`	Logico; se `TRUE` (default) retorna um boxplot comparativo como atributo `"plot"` do resultado.

Details

O teste escolhido por `test` e aplicado por grupo (`by`). Para `"wilcoxon"` usa `stats::wilcox.test()` (nao-parametrico, recomendado para dados ambientais); para `"t_test"` usa `stats::t.test()`; para `"kruskal"` usa `stats::kruskal.test()`.

Value

Um tibble com uma linha por grupo (colunas `by`), contendo:

n_total Total de amostras com valor valido no grupo.

n_chuvoso, n_seco Amostras por periodo.

median_chuvoso, median_seco Medianas por periodo.

mean_chuvoso, mean_seco Medias por periodo.

statistic Estatistica do teste.

p_value P-valor do teste.

diferenca_significativa Logical; `p_value < alpha`.

tendencia Character: `"chuvoso_maior"`, `"seco_maior"` ou `"sem_diferenca"`.

Se `plot = TRUE`, o atributo `attr(resultado, "plot")` contem um objeto `ggplot`.

See Also

`assign_season()`

Other seasonal-tools: `assign_season()`

Examples

```
data("wq_demo", package = "tikatuwq")
d <- assign_season(wq_demo, region = "bahia")
res <- compare_seasons(d, param = "turbidez", by = "ponto", test = "wilcoxon")
print(res)
```

---

compute_load *Carga poluidora (concentracao x vazao)*

Description

Calcula a carga poluidora diaria (ou na unidade desejada) como o produto entre concentracao e vazao: $L = C \times Q \times f$, onde f e o fator de conversao de unidades.

Usage

```
compute_load(
  df,
  param,
  flow_col = "vazao",
  unit_out = c("kg_dia", "t_dia", "kg_ano", "g_s"),
  unit_factor = NULL,
  col_name = NULL
)
```

Arguments

df	Data frame com ao menos a coluna de param e de flow_col.
param	Character; nome da coluna de concentracao (mg/L).
flow_col	Character; nome da coluna de vazao. Default "vazao".
unit_out	Character; unidade de saida. Uma de "kg_dia" (default), "t_dia", "kg_ano", "g_s".
unit_factor	Numeric; fator de conversao personalizado. Se fornecido, sobrepoe unit_out.
col_name	Character; nome da coluna de saida. Default: composto automaticamente como "{param}_carga_{unit_out}".

Details

Unidades de saida suportadas:

"kg_dia" mg/L * m3/s -> kg/dia (fator = 86.4).

"t_dia" mg/L * m3/s -> t/dia (fator = 0.0864).

"kg_ano" mg/L * m3/s -> kg/ano (fator = 31536).

"g_s" mg/L * m3/s -> g/s (fator = 1).

As unidades pressupõem vazao em m³/s e concentracao em mg/L. Ajuste o fator com unit_factor se necessario.

Value

O df de entrada com a nova coluna de carga adicionada.

See Also

exceedance_prob()

Other load-tools: exceedance_prob()

Examples

```
data("wq_demo", package = "tikatuwq")
# Adiciona vazao ficticia para demonstracao
d <- wq_demo
d$vazao <- runif(nrow(d), 2, 10)
d <- compute_load(d, param = "p_total", flow_col = "vazao", unit_out = "kg_dia")
head(d[, c("ponto","p_total","vazao","p_total_carga_kg_dia")])
```

---

conama_check *CONAMA conformity check (detailed; default class = "2")*

Description

For each parameter present in `df`, adds columns:

- `*_ok` (logical),
- `*_status` one of `"ok"`, `"acima_do_maximo"`, `"abaixo_do_minimo"`,
- `*__lim_min` and `*__lim_max` (thresholds used),
- `*__delta` (difference to the relevant limit; >0 above max, <0 below min, 0 if ok).

Usage

```
conama_check(
  df,
  classe = "2",
  environment = c("all", "lotic", "lentic", "intermediate")
)
```

Arguments

`df`	A tibble/data.frame with parameter columns (e.g., ph, turbidez, od, dbo).
`classe`	Character class label (e.g., "especial", "1", "2", "3", "4").
`environment`	Optional aquatic environment context: `"all"` (default), `"lotic"` (rivers), `"lentic"` (lakes/reservoirs), or `"intermediate"`. Used to select specific regulatory thresholds for total phosphorus.

Value

The input `df` with additional columns per parameter as described.

See Also

`conama_limits()`, `conama_summary()`, `conama_report()`, `conama_text()`

Examples

```
## Not run:
data("wq_demo", package = "tikatuwq")
head(conama_check(wq_demo, classe = "2"))

## End(Not run)
```

`conama_freq_check` *Conformidade CONAMA 357/2005 por frequencia*

Description

Avalia a conformidade com a Resolucao CONAMA 357/2005 pela regra de **frequencia**: um parametro e considerado conforme quando o limite e atendido em pelo menos `threshold` das amostras (padrao 80%). A regra e aplicada apenas quando ha `min_n` ou mais amostras no grupo (padrao 6), conforme Art. 15 da Resolucao.

Usage

```
conama_freq_check(
  df,
  classe = "2",
  by = "ponto",
  date_col = "data",
  min_n = 6L,
  threshold = 0.8
)
```

Arguments

`df`	Data frame com colunas de parametros e ao menos as colunas indicadas em `by` e `date_col`.
`classe`	Character; classe CONAMA 357/2005 (ex.: `"2"`).
`by`	Character vector; colunas de agrupamento (ex.: `c("ponto","rio")`). Pode incluir qualquer coluna categorica do data frame.
`date_col`	Character; nome da coluna de datas usada para extrair o ano. Default `"data"`.
`min_n`	Integer; numero minimo de amostras por grupo para aplicar a regra de frequencia. Default `6`.
`threshold`	Numeric em (0, 1]; fracao minima de conformidade exigida. Default `0.80` (80%).

Details

A verificacao linha-a-linha (`conama_check()`) nao reflete o criterio estatistico da norma. Esta funcao agrupa as amostras por `by` e por ano (extraido de `date_col`) e aplica a regra de frequencia.

Quando `n < min_n`, a coluna `freq_conforme` retorna `NA` e `aplicou_regra` retorna `FALSE` — indicando que nao ha amostras suficientes para o criterio estatistico.

Value

Um tibble com uma linha por combinacao by + ano + parametro, contendo as colunas:

ano Ano extraido de date_col.

parametro Nome do parametro avaliado.

n Total de amostras no grupo.

n_ok Amostras dentro do limite.

pct_ok Percentual de conformidade (0-100).

freq_conforme Logical; TRUE se pct_ok >= threshold*100. NA se n < min_n.

aplicou_regra Logical; TRUE se n >= min_n.

See Also

conama_check(), conama_report()

Other conama-tools: balnear_check()

Examples

```
data("wq_demo", package = "tikatuwq")
# Conformidade por ponto e ano (regra dos 80%, min 3 amostras para este dataset)
conama_freq_check(wq_demo, classe = "2", min_n = 3)
```

---

conama_limits *Limits for Brazilian CONAMA 357/2005*

Description

Returns the parameter limits defined by CONAMA Resolution 357/2005 for a given water-use class.

Usage

```
conama_limits(class)
```

Arguments

class	Integer or character. Target class (e.g., 1, 2, 3, 4 or "special"), according to CONAMA 357/2005.

Value

A tibble/data frame with one row per parameter and regulatory thresholds. Typical columns:

- parametro: parameter name (character, normalized to snake_case)
- classe: class label (character)
- min/max (or equivalents): numeric thresholds (may be NA)
- other metadata columns if present (e.g., unit, criterion)

Examples

```
# Class 2 thresholds (first rows)
head(conama_limits(2))
```

`conama_report` *CONAMA conformity report (table)*

Description

CONAMA conformity report (table)

Usage

```
conama_report(
  df,
  classe = "2",
  only_violations = TRUE,
  pretty = FALSE,
  decimal_mark = ",",
  big_mark = "."
)
```

Arguments

`df`	Input data
`classe`	CONAMA class label (e.g., "2")
`only_violations`	If TRUE, returns only rows with `status != "ok"`
`pretty`	If TRUE, returns formatted numeric columns for display
`decimal_mark`	Decimal separator (default `","`)
`big_mark`	Thousands separator (default `"."`)

Value

A tibble. When `pretty = FALSE`: `parametro`, `valor`, `lim_min`, `lim_max`, `status`, `delta`. When `pretty = TRUE`, numeric columns are formatted as character with "natural" decimals.

See Also

`conama_summary()`, `conama_text()`

Examples

```
## Not run:
data("wq_demo", package = "tikatuwq")
conama_report(wq_demo, classe = "2", only_violations = TRUE)
conama_report(wq_demo, classe = "2", only_violations = TRUE, pretty = TRUE)

## End(Not run)
```

`conama_summary` *CONAMA conformity summary (long format)*

Description

CONAMA conformity summary (long format)

Usage

```
conama_summary(df, classe = "2")
```

Arguments

`df`	Input data
`classe`	CONAMA class label

Value

A tibble with columns: `parametro`, `valor`, `lim_min`, `lim_max`, `status`, `ok`, `delta`.

See Also

`conama_check()`, `conama_report()`, `conama_text()`

Examples

```
## Not run:
data("wq_demo", package = "tikatuwq")
head(conama_summary(wq_demo, classe = "2"))

## End(Not run)
```

conama_text *Text summary of conformity (bulleted, formatted)*

Description

Text summary of conformity (bulleted, formatted)

Usage

```
conama_text(
  df,
  classe = "2",
  only_violations = FALSE,
  decimal_mark = ",",
  big_mark = "."
)
```

Arguments

df	Input data
classe	CONAMA class label
only_violations	If TRUE, list only parameters with violation
decimal_mark	Decimal separator (default ",")
big_mark	Thousands separator (default ".")

Value

Character vector of lines (first line is a header, the rest are bullets).

See Also

conama_summary(), conama_report()

Examples

```
## Not run:
data("wq_demo", package = "tikatuwq")
cat(conama_text(wq_demo, classe = "2"), sep = "\n")

## End(Not run)
```

`exceedance_prob` *Probabilidade de excedencia de um limite*

Description

Calcula a probabilidade empirica de que um parametro de qualidade da agua exceda um determinado valor de referencia (threshold), por grupo. Util para analise de risco ambiental e pre-avaliacao de conformidade.

Usage

```
exceedance_prob(
  df,
  param,
  threshold,
  direction = c("above", "below"),
  by = "ponto",
  conf_level = 0.95
)
```

Arguments

`df`	Data frame com ao menos a coluna do `param` e as colunas indicadas em `by`.
`param`	Character; nome do parametro a avaliar.
`threshold`	Numeric; valor de referencia.
`direction`	`"above"` (default) — excedencia acima do limite (ex.: turbidez, DBO, coliformes); ou `"below"` — excedencia abaixo do limite (ex.: OD, saturacao de oxigenio).
`by`	Character vector; colunas de agrupamento. Default `"ponto"`.
`conf_level`	Numeric; nivel de confianca para o intervalo de confianca de Wilson. Default `0.95`.

Details

A probabilidade e calculada como: $P_{ex} = n_{ex}/n_{valid}$ onde n_{ex} e o numero de amostras que excedem `threshold` e n_{valid} e o total de amostras com valor valido (nao-NA).

Para parametros com limite minimo (ex.: OD), usar `direction = "below"`. Para intervalos (ex.: pH 6-9), chamar duas vezes ou usar `conama_freq_check()`.

Value

Um tibble com uma linha por grupo contendo:

n Total de amostras validas no grupo.

n_excedeu Amostras que excedem o threshold.

prob_excedencia Probabilidade empirica (0-1).

ic_inf, ic_sup Intervalo de confianca de Wilson para a proporcao.

See Also

conama_freq_check()

Other load-tools: compute_load()

Examples

```
data("wq_demo", package = "tikatuwq")
# Probabilidade de turbidez acima de 40 NTU (limite CONAMA classe 1)
exceedance_prob(wq_demo, param = "turbidez", threshold = 40, by = "ponto")

# Probabilidade de OD abaixo de 5 mg/L (limite CONAMA classe 2)
exceedance_prob(wq_demo, param = "od", threshold = 5,
                direction = "below", by = "ponto")
```

---

fix_coords *Fix and validate geographic coordinates (lat/lon)*

Description

Normaliza latitude/longitude quando vierem em graus multiplicados por 1e7 (padrao de alguns exports de GPS) e invalida valores fora dos limites.

Usage

```
fix_coords(df, lat = "lat", lon = "lon", divisor = 1e+07)
```

Arguments

df	data.frame de entrada.
lat	Nome da coluna de latitude (padrao: "lat").
lon	Nome da coluna de longitude (padrao: "lon").
divisor	Se abs(valor) exceder os limites, divide por este numero (padrao 1e7).

Value

O df com lat/lon corrigidos quando presentes.

Examples

```
# d <- data.frame(lat = -155432345, lon = -393212345)
# fix_coords(d)
```

generate_analysis *Generate analytical paragraphs (rule-based)*

Description

Produz 3–5 paragrafos curtos, legiveis por humanos, resumindo a qualidade da agua a partir de IQA/WQI, conformidade com a CONAMA 357/2005 e (opcionalmente) tendencias temporais simples. E **rule-based** (nao usa IA) e aceita metadados opcionais para compor o texto.

Usage

```
generate_analysis(
  df,
  classe_conama = "2",
  incluir_tendencia = TRUE,
  parametros_tendencia = c("turbidez", "od", "pH"),
  contexto = list(rio = NA, periodo = NA, cidade = NA)
)
```

Arguments

df	Data frame contendo ao menos a coluna ponto. Recomenda-se tambem as colunas necessarias para checagens CONAMA e para o calculo do IQA.
classe_conama	Character (ex. "2"). Classe-alvo para a checagem da Resolucao CONAMA 357/2005.
incluir_tendencia	Logical; se TRUE, calcula tendencias lineares simples ao longo do tempo.
parametros_tendencia	Character vector; nomes dos parametros para testar tendencia temporal.
contexto	Lista com metadados opcionais (PT/EN), por exemplo list(rio = "Rio Pardo", periodo = "jan-jun/2025", cidade = "Lencois"). As chaves aceitas sao rio/river, periodo/period, cidade.

Value

Vetor character com 3 a 5 paragrafos analiticos prontos para relatorio.

See Also

iqa(), conama_check()

Examples

```
## Not run:
library(tikatuwq)
data("wq_demo")
txt <- generate_analysis(
```

```
  df = wq_demo,
  classe_conama = "2",
  incluir_tendencia = TRUE,
  parametros_tendencia = c("turbidez","od","pH"),
  contexto = list(rio = "Rio Azul", periodo = "jan-jun/2025")
)
cat(paste(txt, collapse = "\n\n"))

## End(Not run)
```

`iet_carlson` *Trophic State Index (Carlson)*

Description

Computa o indice trofico de Carlson (TSI/IET) a partir de profundidade de disco de Secchi, clorofila-a e fosforo total. Retorna componentes e o IET como media por linha dos componentes disponiveis.

Pode receber um `data.frame` como primeiro argumento (ver Detalhes).

Usage

```
iet_carlson(
  secchi = NULL,
  clorofila = NULL,
  tp = NULL,
  .keep_ids = FALSE,
  add_status = TRUE,
  locale = c("pt", "en"),
  ...
)
```

Arguments

`secchi`	Vetor numerico com profundidade de Secchi (m) **ou** um data.frame contendo colunas `secchi` (m), `clorofila` (ug/L) e `tp` (ug/L) ou `p_total` (mg/L). Se for data.frame, `clorofila` e `tp` devem ser `NULL`.
`clorofila`	Vetor numerico com clorofila-a (ug/L).
`tp`	Vetor numerico com fosforo total (ug/L).
`.keep_ids`	Logico; quando data.frame, vincula colunas de ID comuns (`rio`, `ponto`, `data`, `lat`, `lon`). Padrao `FALSE`.
`add_status`	Logico; se `TRUE` (padrao), adiciona a coluna `TSI_status` com a classificacao qualitativa (Carlson).
`locale`	Idioma de `TSI_status`: `"pt"` (padrao) ou `"en"`.
`...`	Reservado para uso futuro (ignorado).

Details

Formulas implementadas (Carlson 1977), com logaritmo natural:

- `TSI_Secchi = 60 - 14.41 * log(secchi)`
- `TSI_Chla = 9.81 * log(clorofila) + 30.6`
- `TSI_TP = 14.42 * log(tp) + 4.15`

Os coeficientes derivam da definicao TSI = 10 (6 - log2 SD): 10 / ln 2 = 14,43. Ate a v0.10.0 o pacote usava `log10`, o que subestimava o indice (ex.: TP = 48 ug/L dava 28,4 em vez de 60).

Quando um data.frame e fornecido, strings com virgula decimal (ex.: "3,2") ou sinais de desigualdade (ex.: "<0,1") sao convertidas com seguranca. Se existir `p_total` (mg/L) em vez de `tp` (ug/L), e feita conversao interna (`tp = p_total * 1000`).

Os componentes e o IET final sao limitados ao intervalo `[0, 100]` para manter consistencia com as figuras e tabelas do pacote/artigo.

Value

Um data.frame com colunas (quando aplicavel):

- `TSI_Secchi` — componente de Secchi (0-100).
- `TSI_Chla` — componente de clorofila-a (0-100).
- `TSI_TP` — componente de fosforo total (0-100).
- `IET` — indice Carlson agregado (media por linha, 0-100).
- `TSI_status` — classe qualitativa (quando `add_status=TRUE`).

References

Carlson, R. E. (1977). A trophic state index for lakes. Limnology and Oceanography, 22(2), 361-369. doi:10.4319/lo.1977.22.2.0361

See Also

`iet_lamparelli()`, `iqa()`, `conama_check()`

Examples

```
# Vetores
secchi <- c(1.2, 0.8, 0.4)        # m
clorofila <- c(5, 12, 30)         # ug/L
tp <- c(20, 40, 70)               # ug/L
iet_carlson(secchi = secchi, clorofila = clorofila, tp = tp)

# Data frame
# df <- data.frame(secchi = secchi, clorofila = clorofila, p_total = c(0.02, 0.04, 0.07))
# iet_carlson(df)                  # converte p_total -> tp (ug/L)
# iet_carlson(df, .keep_ids = TRUE)
```

`iet_lamparelli` *Trophic State Index (Lamparelli)*

Description

Computa o Indice do Estado Trofico de Lamparelli (2004), como adotado pela CETESB, a partir de fosforo total e clorofila-a. O indice agregado e a media de IET(PT) e IET(CL) disponiveis na linha.

Pode receber um `data.frame` como primeiro argumento (ver Detalhes).

Usage

```
iet_lamparelli(
  tp = NULL,
  chla = NULL,
  sd = NULL,
  ambiente = c("rio", "reservatorio"),
  .keep_ids = FALSE,
  add_status = TRUE,
  locale = c("pt", "en"),
  ...
)
```

Arguments

`tp`	Fosforo total (mg/L) **ou** um data.frame contendo colunas `tp` (ug/L) ou `p_total` (mg/L), `chla` ou `clorofila` (ug/L), e `sd` ou `secchi` (m). Se for data.frame, `chla` e `sd` devem ser `NULL`.
`chla`	Clorofila-a (ug/L).
`sd`	Profundidade do disco de Secchi (m).
`ambiente`	Tipo de ambiente: `"rio"` ou `"reservatorio"`.
`.keep_ids`	Logico; quando data.frame, vincula colunas de ID (`rio`, `ponto`, `data`, `lat`, `lon`). Padrao `FALSE`.
`add_status`	Logico; se `TRUE` (padrao), adiciona a coluna `TSI_status` com a classificacao qualitativa (Lamparelli).
`locale`	Idioma de `TSI_status`: `"pt"` (padrao) ou `"en"`.
`...`	Reservado para uso futuro (ignorado).

Details

Equacoes (CETESB 2023, Apendice D, secao 4, p. 10; PT e CL em ug/L, ln = logaritmo natural):

- Rios: `IET(CL) = 10 (6 - ((-0.7 - 0.6 ln CL) / ln 2)) - 20`; `IET(PT) = 10 (6 - ((0.42 - 0.36 ln PT) / ln 2)) - 20`.
- Reservatorios: `IET(CL) = 10 (6 - ((0.92 - 0.34 ln CL) / ln 2))`; `IET(PT) = 10 (6 - ((1.77 - 0.42 ln PT) / ln 2))`.

Conferidas contra os cortes das Tabelas 5 e 6 da mesma fonte. A CETESB nao usa a transparencia no IET; quando `sd` e informado, `IET_Secchi` e devolvido como componente auxiliar (definicao de Carlson, `10 (6 - ln SD / ln 2)`) e nao entra em `IET_Lamp`. Ate a v0.10.0 a funcao usava `10 + 10 log10(x)` para PT e CL, igual para rio e reservatorio, o que classificava praticamente qualquer amostra como ultraoligotrofica.

Entradas com virgula decimal (ex.: "3,2") ou desigualdades (ex.: "<0,1") sao convertidas com seguranca por helpers internos. Se houver apenas `p_total` (mg/L), e convertida para `tp` (ug/L) via `tp = p_total * 1000`.

Os componentes e o indice agregado sao limitados ao intervalo `[0, 100]` para consistencia com as figuras e tabelas do pacote/artigo.

Value

Um data.frame com colunas (quando aplicavel):

- `IET_TP` — componente de fosforo total (0-100).
- `IET_Chla` — componente de clorofila-a (0-100).
- `IET_Secchi` — componente auxiliar de Secchi (0-100), fora do agregado.
- `IET_Lamp` — indice Lamparelli agregado: media de IET_TP e IET_Chla (0-100).
- `TSI_status` — classe qualitativa (quando `add_status=TRUE`).
- `ambiente` — tipo de ambiente informado.

References

Lamparelli, M. C. (2004). Grau de trofia em corpos d'agua do Estado de Sao Paulo: avaliacao dos metodos de monitoramento. Tese (Doutorado), Instituto de Biociencias, Universidade de Sao Paulo.

CETESB (2023). Apendice D: Indices de Qualidade das Aguas e Criterios de Avaliacao da Qualidade dos Sedimentos. In: Qualidade das Aguas Interiores no Estado de Sao Paulo. Secao 4, pp. 9-11.

See Also

`iet_carlson()`, `iqa()`, `conama_check()`

`iqa` *Water Quality Index (WQI / IQA CETESB)*

Description

Calcula o IQA segundo a metodologia CETESB (Apendice D, 2023): subindices q_i agregados por **media geometrica ponderada**, $IQA = \prod_{i=1}^{9} q_i^{w_i}$.

A CETESB publica os q_i apenas como curvas graficas (Figura 1 do Apendice D). O pacote usa equacoes analiticas por partes que aproximam essas curvas; os pesos, os valores fixos fora do dominio (ex.: DBO > 30 -> 2, OD > 140\ os trechos de DBO (5-15 mg/L), nitrogenio total (10-60

mg/L) e pH (8,5-10) foram corrigidos apos comparacao com as curvas. A curva de variacao de temperatura (usada so com `temperature_method` diferente do padrao) nao acompanha a curva oficial. Veja `system.file("audit", "IQA_AUDITORIA.md", package = "tikatuwq")`.

Usage

```
iqa(
  df,
 method = c("CETESB", "CETESB_equations", "IGAM", "CETESB_legacy_approx", "NSF_approx"),
  altitude_m = 0,
  phosphorus_basis = c("P", "PO4"),
  microbial_type = c("thermotolerant_coliforms", "e_coli"),
  temperature_method = c("cetesb_default", "delta", "reference"),
  temperature_reference = NULL,
  delta_temperature = NULL,
  censor_policy = c("limit", "half_limit", "zero", "na", "preserve"),
  allow_partial = FALSE,
  details = FALSE,
  add_status = TRUE,
  locale = c("pt", "en"),
  pesos = NULL,
  nitrate_unit = NULL,
  ...
)
```

Arguments

`df`	Data frame ou tibble com as colunas de monitoramento. Nomes canonicos reconhecidos: `od`, `solidos_totais` (ou `residuo_total`), `ph`, `dbo`, `coliformes_termotolerantes` (ou `coliformes`), `nt_total`, `p_total`, `turbidez`, `temperatura`.
`method`	Metodo de calculo: • `"CETESB"` (padrao) - equacoes analiticas por partes com media geometrica ponderada. • `"CETESB_equations"` - alias de `"CETESB"`. • `"IGAM"` - reproducao computacional documentada do IQA do IGAM (Minas Gerais), motor independente do CETESB e **nao homologado pelo IGAM**. Ver a secao Metodo IGAM. • `"CETESB_legacy_approx"` - subindices por curvas interpoladas aproximadas (legado). • `"NSF_approx"` - subindices por curvas aproximadas com media aritmetica (legado).
`altitude_m`	Altitude em metros (escalar ou vetor com uma altitude por linha), entre -500 e 9000 m; usada na concentracao de saturacao de OD.
`phosphorus_basis`	Base do fosforo informado: `"P"` (default, aplica o fator 3,066 para PO4) ou `"PO4"`. A curva CETESB e de fosfato (PO4); o fator 3,066 consta no Apendice D 2023, p. 5. Em `method = "IGAM"`, deve ser informado explicitamente quando a entrada for `p_total`.

`microbial_type` Tipo microbiologico informado: `"thermotolerant_coliforms"` (default) ou `"e_coli"` (multiplica por 1,25 para equivaler a termotolerantes, conforme o Apendice D CETESB 2023, p. 5). Em `method = "IGAM"` nao ha fator; o argumento so escolhe a coluna quando ambas existem.

`temperature_method`
Componente termico: `"cetesb_default"` (default, qi = 94 constante; a CETESB usa "o valor correspondente a melhor qualidade", com $q^w = 1,58$, compativel com qi = 94), `"delta"` (usa `delta_temperature` ou a coluna `delta_temperatura`; erro se nenhum for informado) ou `"reference"` (usa `temperatura - temperature_reference`; erro se faltar algum). Os modos `"delta"` e `"reference"` sao **experimentais e nao validados**: a curva de Delta T do pacote nao tem fonte rastreavel e diverge da curva q6 da CETESB. Eles emitem aviso e marcam `"experimental_temperature_curve"` em `attr(, "iqa_nonstandard")`. Delta T < -5 C resulta em `NA` (qi indefinido pela fonte).

`temperature_reference`
Temperatura de referencia (escalar ou um valor por linha) quando `temperature_method = "reference"`.

`delta_temperature`
Variacao de temperatura (escalar ou um valor por linha) quando `temperature_method = "delta"`. Tem precedencia sobre a coluna `delta_temperatura`.

`censor_policy` Politica para dados censurados: `"limit"` (default: <X -> X), `"half_limit"` (<X -> X/2), `"zero"`, `"na"`, ou `"preserve"`.

`allow_partial` Logico; se `FALSE` (padrao), exige os 9 componentes. Se `TRUE`, calcula estimativa parcial (veja a secao Dados ausentes).

`details` Logico; se `TRUE`, anexa no atributo `"components"` a tabela de `iqa_components` com os pesos **efetivos** usados em cada linha, de modo que `prod(qi^wi)` reproduz `IQA_unrounded`.

`add_status` Logico; se `TRUE` (padrao), adiciona a coluna `IQA_status`.

`locale` Idioma dos rotulos qualitativos: `"pt"` (padrao) ou `"en"`.

`pesos` Vetor nomeado de pesos customizados (opcional). Nomes aceitos sao os canonicos ou aliases dos 9 parametros; componentes omitidos mantem o peso CETESB e o conjunto e renormalizado para soma 1. Pesos devem ser finitos e nao negativos. Com pesos customizados o resultado deixa de ser o IQA CETESB (registrado em `attr(, "iqa_nonstandard")`).

`nitrate_unit` Usado apenas por `method = "IGAM"`: unidade da coluna `nitrato`, `"N"` (mg/L N-NO3, convertido por 62,004/14,007) ou `"NO3"` (mg/L NO3). Obrigatorio quando houver nitrato; nao ha unidade presumida.

`...` Apenas o argumento legado `na_rm` (depreciado, use `allow_partial`). `classification` e `scheme` geram erro (removidos na 0.11.0). Outros argumentos geram aviso e sao ignorados.

Value

O data frame de entrada com a coluna numerica `IQA` (0-100, arredondada a 2 casas) e, quando `add_status = TRUE`, a coluna fator `IQA_status`. A classe e atribuida sobre o valor nao arredondado

(IQA_unrounded nos detalhes); o arredondamento e apenas de apresentacao. Atributos de proveniencia: iqa_method, iqa_method_reference, iqa_classification, iqa_weights, iqa_custom_weights, iqa_censor_policy, iqa_phosphorus_basis, iqa_microbial_type, iqa_temperature_method, iqa_altitude_m, iqa_allow_partial, iqa_partial_rows, iqa_complete e iqa_nonstandard.

Metodo x classificacao

method define o motor de calculo completo (parametros, curvas, pesos, politica de ausentes) e a escala de classificacao correspondente: iqa(df) calcula e classifica pela CETESB. Para aplicar outra escala a um IQA ja calculado, use classify_iqa; isso nao transforma o numero em um IQA de outra instituicao.

Mudanca incompativel na 0.11.0: o argumento classification de iqa(), presente na 0.10.0 ("cetesb", "legacy"), foi removido. Passa-lo (assim como scheme) gera erro com a orientacao de migracao: classify_iqa(iqa(df)$IQA, classification = "legacy"). A chamada posicional da 0.10.0 (iqa(df, "CETESB", "legacy")) tambem gera erro de migracao, pois o 3o argumento passou a ser altitude_m.

Dados ausentes

Pela CETESB, sem algum dos nove parametros "o calculo do IQA e inviabilizado" (Apendice D 2023, p. 4). Assim, com allow_partial = FALSE (padrao), essas linhas recebem NA. Com allow_partial = TRUE, os pesos dos parametros disponiveis sao renormalizados proporcionalmente ($w_i / \sum w_{disp}$). Esta e uma estimativa parcial do pacote, nao uma regra institucional: o IGAM, por exemplo, nao calcula o IQA sem OD ou coliformes e reparte o peso dos demais ausentes igualmente. Linhas parciais ficam registradas no atributo "iqa_partial_rows" e, com details = TRUE, nas colunas IQA_partial e missing_params.

Metodo IGAM

method = "IGAM" calcula pelas equacoes do SCQA (SEMAD/UCEMG/PNMA II, 2005) e pelas regras do Projeto Aguas de Minas: pesos 0,17 (OD), 0,15 (microbiologico), 0,12 (pH), 0,10 (DBO, nitrato, fosfato total, temperatura), 0,08 (turbidez, solidos totais); qs de temperatura constante igual a 92; sem OD ou microbiologico o IQA nao e calculado; o peso dos demais ausentes e repartido igualmente. A turbidez usa o coeficiente $1,5$ no termo do cosseno (Bucci 2011; Nunes 2008), em vez do 15 impresso em copias do SCQA. A implementacao reproduz 49/49 IQAs publicados pelo IGAM (sub-bacia da Pampulha, 2017-2018), mas **nao e homologada pelo IGAM** e nao foi validada em outras bacias.

Entradas: od (mg/L), temperatura e cloreto (mg/L; a saturacao de OD do SCQA exige cloreto, sem valor padrao), e_coli ou coliformes_termotolerantes (sem fator de conversao; se ambas existirem, informe microbial_type), ph, dbo, nitrato (exige nitrate_unit), fosfato_total (PO4) ou p_total (exige phosphorus_basis explicito), turbidez e solidos_totais. Nitrogenio total, ortofosfato e TDS nao substituem nitrato, fosfato total e solidos totais. allow_partial, pesos e os argumentos de temperatura do CETESB nao se aplicam (erro). O IQA e arredondado a 1 casa (como o IGAM publica) e a classe IGAM e atribuida sobre esse valor; details = TRUE traz o valor bruto, os qi e os pesos efetivos. Nota completa: system.file("methodology","IGAM_VALIDATION.md", package = "tikatuwq").

Valores invalidos

Concentracoes negativas, pH fora de [0, 14], temperatura fora de [-2, 50] C e valores nao finitos sao convertidos em NA com aviso (e nao truncados para zero).

Examples

```
d <- data.frame(
  od = 6.5, temperatura = 25, dbo = 2, coliformes = 200,
  ph = 7.2, nt_total = 1.2, p_total = 0.05, turbidez = 8,
  solidos_totais = 120
)
out <- iqa(d)
out$IQA
out$IQA_status

# Outra escala para o mesmo numero: use classify_iqa()
classify_iqa(out$IQA, classification = "igam")

# Metodos disponiveis e escopo de validacao
iqa_methods()

# IQA pelo metodo IGAM (reproducao documentada, nao homologada pelo IGAM).
# Amostra sintetica; as unidades de nitrato (mg/L N) e de fosforo total
# (mg/L P) sao declaradas explicitamente.
g <- data.frame(
  od = 7, temperatura = 25, cloreto = 10, e_coli = 100, ph = 7.3,
  dbo = 2, nitrato = 1, p_total = 0.1, turbidez = 5, solidos_totais = 100
)
iqa(g, method = "IGAM", altitude_m = 0, nitrate_unit = "N",
    phosphorus_basis = "P")   # IQA 75.5, "Bom"

# Auditoria: pesos efetivos e termos por componente
comps <- attr(iqa(d, details = TRUE), "components")
prod(unlist(comps[1, grep("^term_", names(comps))]))
```

---

iqa_components *Auditoria detalhada dos subindices do IQA (metodo CETESB)*

Description

Retorna uma tabela estruturada contendo o valor bruto, valor transformado, subindice Qi, peso Wi e parcela ponderada Qi^{Wi} para cada um dos 9 componentes oficiais do IQA CETESB.

Usage

```
iqa_components(
  df,
  altitude_m = 0,
  phosphorus_basis = c("P", "PO4"),
```

```
  microbial_type = c("thermotolerant_coliforms", "e_coli"),
  temperature_method = c("cetesb_default", "delta", "reference"),
  temperature_reference = NULL,
  delta_temperature = NULL,
  censor_policy = c("limit", "half_limit", "zero", "na", "preserve"),
  method = c("CETESB", "IGAM")
)
```

Arguments

`df`	Data frame de entrada com os dados de monitoramento.
`altitude_m`	Altitude em metros (default 0).
`phosphorus_basis`	"P" (default) ou "PO4".
`microbial_type`	"thermotolerant_coliforms" (default) ou "e_coli".
`temperature_method`	"cetesb_default" (default), "delta" ou "reference".
`temperature_reference`	Vetor ou escalar com temperatura de referencia.
`delta_temperature`	Vetor ou escalar com a variacao de temperatura (Delta T em graus Celsius).
`censor_policy`	Politica de tratamento de dados censurados ("limit", "half_limit", "zero", "na", "preserve").
`method`	Motor de calculo: `"CETESB"` (padrao) ou `"IGAM"`. Para o IGAM os componentes sao obtidos com `iqa(df, method = "IGAM", details = TRUE)`; aqui `"IGAM"` gera erro com essa orientacao.

Value

Um tibble ou data frame detalhado com colunas para cada parametro e o IQA calculado.

Examples

```
d <- data.frame(
  od = 6.5, temperatura = 25, dbo = 2, coliformes = 200,
  ph = 7.2, nt_total = 1.2, p_total = 0.05, turbidez = 8,
  solidos_totais = 120
)
iqa_components(d)
```

iqa_methods *Metodos de calculo do IQA disponiveis no pacote*

Description

Lista os motores de calculo do IQA, sua situacao de validacao e a escala de classificacao que cada um aplica por padrao. Use para verificar, antes de calcular, se um metodo esta disponivel.

Usage

```
iqa_methods()
```

Details

- `"CETESB"`: padrao. Equacoes que aproximam as curvas da CETESB (Apendice D 2023); ver `system.file("audit", "VALIDACAO_CETESB.md", package = "tikatuwq")`.
- `"IGAM"`: reproducao computacional documentada do IQA do IGAM (equacoes do SCQA, SEMAD 2005; regras do Projeto Aguas de Minas), com a turbidez em `k = 1,5` (Bucci 2011; Nunes 2008). Reproduz 49/49 IQAs publicados pelo IGAM na sub-bacia da Pampulha (2017-2018). **Nao e homologada pelo IGAM.** Ver `system.file("methodology", "IGAM_VALIDATION.md", package = "tikatuwq")`.
- `"CETESB_legacy_approx"` e `"NSF_approx"`: rotas legadas por curvas interpoladas aproximadas.

A ANA nao tem metodo proprio no pacote: o portal da ANA usa os parametros e pesos da CETESB, mas a equivalencia das curvas nao foi verificada (ver `system.file("audit", "ANA_VS_CETESB.md", package = "tikatuwq")`).

Value

Data frame com as colunas `method`, `implemented` (motor presente no pacote), `documented` (fontes registradas), `reference_validated`, `available` (pode ser usado em `iqa()`), `classification`, `limitation` e `reference`.

Semantica de `reference_validated`: `TRUE` significa que o motor reproduz valores de IQA *publicados* pela fonte de referencia (exemplo resolvido, no caso CETESB; IQAs por amostra publicados pelo IGAM, no caso IGAM), com o escopo descrito em `limitation`. **Nao** significa homologacao, certificacao ou endosso da instituicao, nem validacao em todas as bacias ou condicoes analiticas.

Examples

```
iqa_methods()
```

`mk_seasonal` *Teste de Mann-Kendall sazonal*

Description

Implementa o teste de Mann-Kendall sazonal de Hirsch, Slack & Smith (1982), adequado para series temporais de qualidade da agua com sazonalidade (periodos chuvoso/seco ou meses do ano). A estatistica S total e a soma das estatisticas S computadas separadamente por estacao, o que remove o vies introduzido pela sazonalidade nos testes convencionais.

Usage

```
mk_seasonal(
  df,
  param,
  date_col = "data",
  by = "ponto",
  period = c("monthly", "season"),
  season_col = "season",
  alpha = 0.05,
  locale = c("pt", "en")
)
```

Arguments

`df`	Data frame com ao menos a coluna do `param` e `date_col`.
`param`	Character; nome da coluna do parametro a testar.
`date_col`	Character; nome da coluna de datas. Default `"data"`.
`by`	Character vector; colunas de agrupamento. Default `"ponto"`.
`period`	`"monthly"` (default) — 12 estacoes (meses 1–12); ou `"season"` — 2 estacoes definidas por `season_col` (requer `assign_season()` previamente).
`season_col`	Character; nome da coluna de periodo hidrologico, usado apenas quando `period = "season"`. Default `"season"`.
`alpha`	Numeric; nivel de significancia. Default `0.05`.
`locale`	Character; idioma dos rotulos de tendencia: `"pt"` (default) ou `"en"`.

Details

Algoritmo (Hirsch et al., 1982):

1. Para cada estacao *m*, extrai as observacoes dentro daquela estacao e calcula S_m e $VAR(S_m)$.
2. Soma: $S = \sum S_m$, $VAR(S) = \sum VAR(S_m)$.
3. Estatistica Z com correcao de continuidade: $Z = (S - \text{sgn}(S))/\sqrt{VAR(S)}$.
4. p-valor bilateral pela distribuicao normal padrao.

A inclinacao de Sen e calculada sobre a serie completa (mediana de todas as inclinacoes pareadas $(x_j - x_i)/(t_j - t_i)$), expressa em unidades do parametro por ano.

Requer ao menos 3 observacoes por estacao para incluir a estacao no calculo; estacoes com menos dados sao ignoradas (com aviso).

Value

Um tibble com uma linha por grupo, contendo:

parametro Nome do parametro testado.

n_obs Total de observacoes validas.

n_estacoes Numero de estacoes com dados suficientes (>= 3).

S Estatistica S de Mann-Kendall agregada.

varS Variancia de S.

Z Estatistica Z normalizada.

p_value p-valor bilateral.

tau Tau de Kendall normalizado.

sen_slope Inclinacao de Sen (unidade/ano).

significativo Logical; `p_value < alpha`.

tendencia `"crescente"`, `"decrescente"` ou `"sem_tendencia"`.

References

Hirsch, R. M., Slack, J. R., & Smith, R. A. (1982). Techniques of trend analysis for monthly water quality data. *Water Resources Research*, 18(1), 107–121. doi:10.1029/WR018i001p00107

See Also

`assign_season()`, `trend_param()`

Examples

```
data("wq_demo", package = "tikatuwq")

# Mann-Kendall mensal (period = "monthly")
mk_seasonal(wq_demo, param = "turbidez", by = "ponto")


# Mann-Kendall por periodo hidrologico
d <- assign_season(wq_demo, region = "bahia")
mk_seasonal(d, param = "turbidez", by = "ponto",
            period = "season", season_col = "season")
```

normalize_param_names *Normaliza nomes de colunas de um df para os nomes canonicos do pacote*

Description

Renomeia colunas cujos nomes sejam aliases conhecidos para o nome canonico. Util como primeiro passo em leitura de dados externos.

Usage

```
normalize_param_names(df)
```

Arguments

df	Data frame de entrada.

Value

O df com colunas renomeadas para nomes canonicos quando reconhecidas.

Examples

```
d <- data.frame(pH = 7, BOD = 3, DO = 6)
normalize_param_names(d)
```

nsfwqi *National Sanitation Foundation Water Quality Index (NSF WQI)*

Description

Calcula o NSF WQI suportando a agregacao multiplicativa (McClelland / NSF revisado: $WQI = \prod Q_i^{w_i}$) e a agregacao aditiva original (Brown et al., 1970: $WQI = \sum Q_i \cdot w_i$).

Usage

```
nsfwqi(
  df,
  pesos = c(do = 0.17, fc = 0.16, ph = 0.11, bod = 0.11, temp_change = 0.1, po4 = 0.1,
    no3 = 0.1, turbidez = 0.08, ts = 0.07),
  aggregation = c("mcclelland_geometric", "brown1970_arithmetic"),
  na_rm = FALSE,
  add_status = TRUE,
  locale = c("pt", "en")
)
```

Arguments

df	Data frame com as colunas dos parametros.
pesos	Vetor nomeado numerico com os pesos oficiais NSF: do=0.17, fc=0.16, ph=0.11, bod=0.11, temp_change=0.10, po4=0.10, no3=0.10, turbidez=0.08, ts=0.07.
aggregation	Metodo de agregacao: "mcclelland_geometric" (padrao geometrico ponderado) ou "brown1970_arithmetic" (padrao aritmetico ponderado original de 1970).
na_rm	Logico; se TRUE, renormaliza os pesos por linha para os parametros validos. Se FALSE (padrao), linhas com parametros faltantes resultam em NA.
add_status	Logico; se TRUE (padrao), adiciona a coluna NSFWQI_status.
locale	Idioma dos rotulos: "pt" (padrao) ou "en".

Details

O mapeamento de parametros tenta resolver colunas em formato brasileiro ou internacional:

do Oxigenio dissolvido (mg/L ou sat).

fc Coliformes termotolerantes / fecais (NMP/100 mL).

ph pH (unidades).

bod Demanda bioquimica de oxigenio (mg/L).

temp_change Variacao termica Delta T (graus Celsius).

po4 Fosfato total (mg/L).

no3 Nitrato (mg/L).

turbidez Turbidez (NTU).

ts Solidos totais (mg/L). Solidos suspensos (TSS) nao sao usados como substituto (ate a v0.11.0 eram, sem aviso).

Value

O data frame de entrada com a coluna NSFWQI e, opcionalmente, NSFWQI_status.

Natureza aproximada

Nao e o NSF WQI validado. Os subindices Q_i sao funcoes-degrau grosseiras, com 2 a 4 niveis (ex.: OD >= 7,5 mg/L -> 90), e nao as curvas da NSF (Brown et al., 1970). Consequencias:

- o Q_i maximo e 90, logo o indice nunca passa de 90 e a classe "Excelente" (> 90) e inalcancavel;
- o OD e usado em mg/L, enquanto a NSF usa \
- sem coluna de variacao de temperatura, assume-se Delta T = 0 (Q = 90).

Use apenas para exploracao e comparacao qualitativa. Os pesos padrao sao os pesos NSF publicados. Ver system.file("audit","RELATORIO_IMPLEMENTACAO_IGAM_COMPARE.md", package = "tikatuwq").

References

Brown, R. M., McClelland, N. I., Deininger, R. A., & Tozer, R. G. (1970). A water quality index - do we dare? *Water and Sewage Works*, 117(10), 339-343.

McClelland, N. I. (1974). *Water Quality Index Application in the Kansas River Basin*. EPA-907/9-74-001, U.S. Environmental Protection Agency.

Examples

```
d <- data.frame(
  od = 6.5, coliformes = 200, ph = 7.2, dbo = 2,
  temperatura = 25, p_ortofosfato = 0.05, n_nitrato = 1.0,
  turbidez = 8, solidos_totais = 120
)
nsfwqi(d, na_rm = TRUE)
```

`param_plot` *Plot temporal de um parametro (com filtro por rio e/ou ponto)*

Description

Gera grafico temporal para **um parametro**, com opcoes de filtro por `rios` e/ou `pontos`. Se houver mais de um ponto, a cor diferencia pontos; opcional `facet = TRUE` para facetar por ponto. Pode adicionar reta de tendencia com `add_trend = TRUE` (lm).

Usage

```
param_plot(
  df,
  parametro,
  rios = NULL,
  pontos = NULL,
  add_trend = TRUE,
  facet = FALSE
)
```

Arguments

`df`	Data frame com `data` e a coluna do `parametro`. Idealmente contem `ponto`, e opcionalmente `rio`.
`parametro`	Character; nome do parametro.
`rios`	Vetor de nomes de rio a filtrar (opcional; usa coluna `rio` se existir).
`pontos`	Vetor de pontos a filtrar (opcional; usa coluna `ponto` se existir).
`add_trend`	Logical; se `TRUE`, adiciona `geom_smooth(method = "lm", se = FALSE)`.
`facet`	Logical; se `TRUE` e houver `ponto`, aplica `facet_wrap(~ponto)`.

Value

Objeto ggplot.

See Also

Other parameter-tools: param_plot_multi(), param_summary(), param_summary_multi(), param_trend(), param_trend_multi()

Examples

```
## Not run:
data("wq_demo", package = "tikatuwq")
param_plot(wq_demo, "turbidez", pontos = c("P1","P2"), add_trend = TRUE, facet = TRUE)

## End(Not run)
```

---

param_plot_multi *Plot temporal para varios parametros (filtro por rio/ponto)*

Description

Combina varios parametros em um unico grafico. Por padrao:

- cor = ponto (se existir);
- facet = "parametro" cria paineis por parametro;
- facet = "grid" usa grade ponto ~ parametro quando ha mais de um ponto.

Usage

```
param_plot_multi(
  df,
  parametros,
  rios = NULL,
  pontos = NULL,
  add_trend = TRUE,
  facet = c("parametro", "none", "grid")
)
```

Arguments

df	Data frame com data e colunas dos parametros.
parametros	Vetor de nomes de parametros.
rios	Vetor de rios para filtrar (opcional; usa coluna rio se existir).
pontos	Vetor de pontos para filtrar (opcional; usa coluna ponto se existir).
add_trend	Logical; se TRUE, adiciona reta lm em cada painel.
facet	"parametro", "none" ou "grid".

Value

Objeto ggplot.

See Also

Other parameter-tools: param_plot(), param_summary(), param_summary_multi(), param_trend(), param_trend_multi()

Examples

```
## Not run:
data("wq_demo", package = "tikatuwq")
param_plot_multi(wq_demo, c("turbidez","od"), pontos = c("P1","P2"),
                 add_trend = TRUE, facet = "grid")

## End(Not run)
```

---

param_summary *Resumo estatistico por parametro (com filtro por rio e/ou ponto)*

Description

Produz resumo estatistico para **um parametro**, com opcoes de filtro por rios e/ou pontos e agregacao opcional por periodo (mes/trimestre/ano), quando houver coluna data.

Usage

```
param_summary(
  df,
  parametro,
  rios = NULL,
  pontos = NULL,
  period = c("none", "month", "quarter", "year"),
  na_rm = TRUE
)
```

Arguments

df	Data frame com ao menos a coluna do parametro. Idealmente contem ponto, e opcionalmente rio e data.
parametro	Character; nome do parametro (ex.: "turbidez", "od", "pH").
rios	Vetor de nomes de rio a filtrar (opcional; usa coluna rio se existir).
pontos	Vetor de pontos a filtrar (opcional; usa coluna ponto se existir).
period	"none", "month", "quarter" ou "year" para agregar por periodo (requer coluna data).
na_rm	Remover NA dos calculos? (default TRUE)

Value

Tibble com colunas de agrupamento disponiveis (rio, ponto, periodo) e metricas: `n`, `mean`, `sd`, `min`, `median`, `max`.

See Also

Other parameter-tools: `param_plot()`, `param_plot_multi()`, `param_summary_multi()`, `param_trend()`, `param_trend_multi()`

Examples

```
## Not run:
data("wq_demo", package = "tikatuwq")
param_summary(wq_demo, "turbidez", pontos = "P1")
param_summary(wq_demo, "od", rios = "Rio Azul", period = "month")

## End(Not run)
```

`param_summary_multi` *Resumo para varios parametros (filtro por rio/ponto)*

Description

Itera sobre um vetor de `parametros`, chamando `param_summary()` para cada um, e combina as saidas em uma unica tabela, acrescentando a coluna `parametro`.

Usage

```
param_summary_multi(
  df,
  parametros,
  rios = NULL,
  pontos = NULL,
  period = c("none", "month", "quarter", "year"),
  na_rm = TRUE
)
```

Arguments

`df`	Data frame com colunas necessarias (ver `param_summary()`).
`parametros`	Vetor de nomes de parametros (ex.: c("turbidez","od","pH")).
`rios`	Vetor de rios para filtrar (opcional; usa coluna `rio` se existir).
`pontos`	Vetor de pontos para filtrar (opcional; usa coluna `ponto` se existir).
`period`	"none","month","quarter","year" (igual a `param_summary()`).
`na_rm`	Logical; repassado para `param_summary()`.

Value

Tibble combinando os resumos de todos os parametros, com coluna `parametro` indicando a origem.

See Also

Other parameter-tools: `param_plot()`, `param_plot_multi()`, `param_summary()`, `param_trend()`, `param_trend_multi()`

Examples

```
## Not run:
data("wq_demo", package = "tikatuwq")
param_summary_multi(wq_demo, c("turbidez","od"), pontos = c("P1","P2"))

## End(Not run)
```

`param_trend` *Tendencia temporal por parametro (por rio/ponto se existentes)*

Description

Ajusta um modelo **lm(valor ~ tempo)** para **um parametro**, retornando `slope`, `p_value`, `r2` e `n`. Se existirem colunas `rio` e/ou `ponto`, calcula por grupo; caso contrario, calcula geral.

Usage

```
param_trend(df, parametro, rios = NULL, pontos = NULL, na_rm = TRUE)
```

Arguments

`df`	Data frame com `data` e a coluna do `parametro`. Idealmente contem `ponto`, e opcionalmente `rio`.
`parametro`	Character; nome do parametro.
`rios`	Vetor de nomes de rio a filtrar (opcional; usa coluna `rio` se existir).
`pontos`	Vetor de pontos a filtrar (opcional; usa coluna `ponto` se existir).
`na_rm`	Remover `NA` antes do ajuste? (default `TRUE`)

Value

Tibble com colunas de agrupamento (quando existirem) + `slope` (por dia), `p_value`, `r2`, `n`.

See Also

Other parameter-tools: `param_plot()`, `param_plot_multi()`, `param_summary()`, `param_summary_multi()`, `param_trend_multi()`

Examples

```
## Not run:
data("wq_demo", package = "tikatuwq")
param_trend(wq_demo, "turbidez", pontos = c("P1","P2"))

## End(Not run)
```

`param_trend_multi` *Tendencia para varios parametros (filtro por rio/ponto)*

Description

Itera sobre um vetor de `parametros`, chamando `param_trend()` para cada um, e combina as saidas em uma unica tabela, acrescentando a coluna `parametro`.

Usage

```
param_trend_multi(df, parametros, rios = NULL, pontos = NULL, na_rm = TRUE)
```

Arguments

`df`	Data frame com `data` e colunas dos parametros.
`parametros`	Vetor de nomes de parametros.
`rios`	Vetor de rios (opcional; usa coluna `rio` se existir).
`pontos`	Vetor de pontos (opcional; usa coluna `ponto` se existir).
`na_rm`	Logical; repassado para `param_trend()`.

Value

Tibble combinando as tendencias de todos os parametros, com coluna `parametro`.

See Also

Other parameter-tools: `param_plot()`, `param_plot_multi()`, `param_summary()`, `param_summary_multi()`, `param_trend()`

Examples

```
## Not run:
data("wq_demo", package = "tikatuwq")
param_trend_multi(wq_demo, c("turbidez","od"), pontos = "P1")

## End(Not run)
```

plot_box *Boxplots by site/parameter*

Description

Boxplots of one numeric parameter grouped by a categorical column.

Usage

```
plot_box(df, parametro, by = "ponto")
```

Arguments

df	Data frame with water quality data.
parametro	Character; name of the numeric parameter column.
by	Character; grouping column (e.g., "ponto").

Value

A ggplot object.

See Also

plot_series(), plot_heatmap(), iqa()

Examples

```
data(wq_demo)
plot_box(wq_demo, "turbidez", by = "ponto")
```

plot_heatmap *Heatmap of parameters vs. sites*

Description

Heatmap for long-format data (date x parameter).

Usage

```
plot_heatmap(df_long)
```

Arguments

df_long	Long-format data frame with columns data, parametro, valor.

Value

A ggplot object.

Examples

```
# Example: reshape wq_demo to long and plot
data(wq_demo)
library(tidyr)
df_long <- tidyr::pivot_longer(
  wq_demo,
  cols = c("ph","od","turbidez","dbo"),
  names_to = "parametro",
  values_to = "valor"
)
plot_heatmap(df_long)
```

---

plot_iet *Visualiza o Indice de Estado Trofico (IET / TSI)*

Description

Grafico de barras horizontais ou colunas verticais para o IET/TSI, com coloracao por classe trofica. Aceita resultados das funcoes `iet_carlson()` ou `iet_lamparelli()`.

Usage

```
plot_iet(
  df,
  iet_col = NULL,
  method = c("carlson", "lamparelli"),
  orientation = c("vertical", "horizontal"),
  facet = NULL
)
```

Arguments

`df`	Data frame retornado por `iet_carlson()` ou `iet_lamparelli()`, ou qualquer data frame com uma coluna de IET numerica e a coluna `ponto`.
`iet_col`	Character; nome da coluna de IET. Se `NULL` (default), detectada automaticamente.
`method`	Character; metodo de classificacao trofica: `"carlson"` (default) ou `"lamparelli"`.
`orientation`	Character; `"vertical"` (default) ou `"horizontal"`.
`facet`	Character ou `NULL`; coluna para facetar (ex.: `"rio"`). Default `NULL`.

Details

A funcao detecta automaticamente a coluna de IET: procura por "IET", "TSI", "IET_Carlson" ou "IET_Lamparelli". Tambem e possivel especificar o nome via iet_col.

As faixas de classificacao trofica seguem o metodo escolhido:

- Carlson (1977): Ultraoligo (<30), Oligo (30-40), Meso (40-50), Eutro (50-70), Hipereutro (>70).
- Lamparelli (2004): Ultraoligo (<47), Oligo (47-52), Meso (52-59), Eutro (59-63), Supereutro (63-67), Hipereutro (>67).

Value

Um objeto ggplot.

See Also

iet_carlson(), iet_lamparelli(), plot_iqa()

Other visualization-tools: plot_map_quality()

Examples

```
data("wq_demo", package = "tikatuwq")
df_iet <- iet_carlson(wq_demo, .keep_ids = TRUE)
plot_iet(df_iet, method = "carlson")

df_lamp <- iet_lamparelli(wq_demo, ambiente = "rio", .keep_ids = TRUE)
plot_iet(df_lamp, method = "lamparelli")
```

---

plot_iqa *Plot IQA by site/date*

Description

Bar plot of IQA values per site/date. Requires an IQA column.

Usage

```
plot_iqa(df)
```

Arguments

df	Data frame returned by iqa() (or with equivalent columns).

Value

A ggplot object.

See Also

iqa(), plot_series(), plot_box()

Examples

```
data(wq_demo)
d <- iqa(wq_demo, allow_partial = TRUE)
plot_iqa(d)
```

`plot_map` *Plot interactive map of sampling points (default Leaflet pins)*

Description

Creates an interactive Leaflet map of sampling points using the **default Leaflet marker** (blue pin). Latitude/longitude are autodetected from columns `lat` and `lon`. If these columns are not present, but `latitude` and/or `longitude` exist, they are automatically copied to `lat` and `lon`. You may group layers with `group_by` (e.g., "year") and show popups with `popup`.

If `color_by` is provided, a legend is drawn to describe the values, but **markers are not colorized** (the default *Leaflet* pin has fixed style).

Usage

```
plot_map(
  df,
  color_by = NULL,
  popup = NULL,
  group_by = NULL,
  legend_title = NULL,
  na_rm = TRUE
)
```

Arguments

`df`	data.frame/tibble with coordinates; must contain `lat`/`lon` (or `latitude`/`longitude`, which will be mapped automatically).
`color_by`	optional column used to build a legend (numeric or factor). It does not change the marker color.
`popup`	optional column name with popup/tooltip text.
`group_by`	optional column name to create overlay layers (e.g., "year").
`legend_title`	optional legend title (used when `color_by` is set).
`na_rm`	logical; if `TRUE` (default) remove rows with invalid coordinates.

Details

The function expects coordinates in columns named `lat` and `lon`. If those columns are not found, but `latitude` and/or `longitude` are present, they are copied to `lat` and `lon` respectively before plotting.

Value

a `leaflet` htmlwidget.

Examples

```
data("wq_demo", package = "tikatuwq")
d2 <- wq_demo |>
  validate_wq() |>
  iqa(allow_partial = TRUE)
d2$year <- as.integer(format(d2$data, "%Y"))

# Marcadores padrao + legenda de IQA
plot_map(d2, color_by = "IQA", group_by = "year", popup = "ponto",
         legend_title = "IQA (0-100)")
```

---

plot_map_quality *Mapa interativo de qualidade da agua por ponto de amostragem*

Description

Cria um mapa Leaflet interativo colorindo cada ponto de amostragem de acordo com o valor do indice de qualidade da agua calculado (IQA, IET ou NSF WQI). Util para identificar espacialmente pontos criticos e gradientes de qualidade ao longo de uma bacia.

Usage

```
plot_map_quality(
  df,
  index_col = NULL,
  index = c("IQA", "IET_carlson", "IET_lamparelli", "NSFWQI"),
  lat_col = "lat",
  lon_col = "lon",
  label_col = "ponto",
  radius = 10,
  locale = c("pt", "en")
)
```

Arguments

df	Data frame com ao menos as colunas de coordenadas (lat_col, lon_col) e o indice de qualidade.
index_col	Character; nome da coluna do indice. Se NULL (default), detectada automaticamente.
index	Character; nome do indice para determinacao da paleta: "IQA" (default), "IET_carlson", "IET_lamparelli" ou "NSFWQI". Ignorado quando index_col esta especificado e o nome da coluna identifica o indice univocamente.
lat_col	Character; coluna de latitude. Default "lat".
lon_col	Character; coluna de longitude. Default "lon".
label_col	Character; coluna para rotulo do popup. Default "ponto".
radius	Numeric; raio dos circulos em pixels. Default 10.
locale	Character; idioma dos rotulos de classificacao: "pt" (default) ou "en".

Details

A funcao detecta automaticamente a coluna de indice quando index_col = NULL, procurando por: "IQA", "IET", "IET_Carlson", "IET_Lamparelli", "TSI", "NSFWQI".

Requer o pacote **leaflet** (listado em Suggests). Se nao estiver instalado, a funcao emite uma mensagem e retorna invisible(NULL).

Quando o data frame tiver multiplas linhas por ponto, e usada a **media** do indice por ponto para representacao no mapa.

Value

Um objeto leaflet (HTML widget) ou invisible(NULL) se **leaflet** nao estiver disponivel.

See Also

plot_map(), iqa(), iet_carlson()

Other visualization-tools: plot_iet()

Examples

```
if (requireNamespace("leaflet", quietly = TRUE)) {
  data("wq_demo", package = "tikatuwq")
  d <- iqa(wq_demo, allow_partial = TRUE)
  plot_map_quality(d, index = "IQA")
}
```

plot_series *Time series by parameter*

Description

Plot a time series for one numeric parameter, optionally colored/faceted by a grouping column.

Usage

```
plot_series(df, parametro, facet = NULL)
```

Arguments

df	Data frame with a data column (Date/POSIXct) and the parameter column.
parametro	Character; name of the numeric column to plot on Y.
facet	Character or NULL; optional grouping column name to color/facet.

Value

A ggplot object.

See Also

plot_box(), plot_heatmap(), iqa()

Examples

```
data(wq_demo)
# Basic: time series of turbidity
p <- plot_series(wq_demo, "turbidez")
# With color/facet by sampling point
p2 <- plot_series(wq_demo, "turbidez", facet = "ponto")
```

plot_trend *Linha de tendencia temporal para parametros de qualidade da agua*

Description

Gera um grafico de series temporais com pontos observados e linhas de tendencia ajustadas. Suporta metodos robustos (Theil-Sen), lineares (OLS) ou suavizados (LOESS). Util para verificar tendencias de parametros ambientais por ponto e/ou rio.

Usage

```
plot_trend(
  data,
  param,
  date_col = "data",
  group_cols = c("rio", "ponto"),
  method = c("theilsen", "ols", "loess"),
  show_points = TRUE,
  min_n = 6
)
```

Arguments

data	data.frame. Deve conter ao menos uma coluna de datas e a coluna do parametro a ser analisado.
param	character. Nome da coluna do parametro (ex.: "turbidez", "iqa").
date_col	character. Nome da coluna de datas. Default = "data".
group_cols	character. Vetor com colunas para agrupamento (ex.: c("rio","ponto")). Use "none" para nao facetar. Default = c("rio","ponto").
method	character. Metodo de ajuste da tendencia: • "theilsen" (padrao): regressao Theil-Sen (robusta a outliers). • "ols": regressao linear simples (minimos quadrados). • "loess": curva suavizada, sem inclinacao unica.
show_points	logical. Mostrar pontos observados? Default = TRUE.
min_n	integer. Numero minimo de observacoes por grupo para calcular tendencia. Default = 6.

Details

- A funcao desenha pontos e linhas conectando as observacoes, alem da linha de tendencia calculada pelo metodo escolhido.
- Quando group_cols possui mais de uma categoria, os grupos sao facetados.
- "theilsen" e mais robusto a valores atipicos do que "ols".
- "loess" e util quando nao se espera relacao linear no tempo.

Value

Objeto ggplot2, que pode ser plotado diretamente.

See Also

plot_series(), iqa()

Examples

```
# Exemplo simples: turbidez com tendencia Theil-Sen
set.seed(1)
df <- data.frame(
  data = as.Date("2024-01-01") + 0:11*30,
  rio = "Demo", ponto = "P1",
  turbidez = 20 + (-0.3)*(0:11) + rnorm(12, 0, 1)
)
plot_trend(df, param = "turbidez", method = "theilsen")

# Exemplo com multiplos grupos e facetamento (OLS)
df2 <- data.frame(
  data = rep(seq(as.Date("2024-01-01"), by = "30 days", length.out = 12), 2),
  rio = rep(c("Rio A","Rio B"), each = 12),
  ponto = rep(c("P1","P2"), each = 12),
  od = c(7 + rnorm(12, 0, 0.5), 6 + rnorm(12, 0, 0.5))
)
plot_trend(df2, param = "od", method = "ols")
```

`read_wq` *Read water-quality CSV (robust parsing)*

Description

Le um CSV com **delimitador virgula ou ponto-e-virgula** e **virgula ou ponto** como separador decimal, ignorando sufixos de unidade (ex.: `"0,04 mg/L"`). Le tudo como texto primeiro, normaliza nomes, e faz parse robusto de colunas numericas. Ajusta pH evidentemente fora de faixa (ex.: `72 -> 7.2`). Opcionalmente normaliza coordenadas geograficas se vierem em graus * 1e7.

Usage

```
read_wq(
  path,
  tz = "America/Bahia",
  normalize_coords = TRUE,
  nd_policy = c("ld2", "ld", "zero", "na")
)
```

Arguments

`path`	Caminho para o arquivo CSV.
`tz`	Fuso horario para datas (mantido por compatibilidade; datas sao `Date`).
`normalize_coords`	Logico; se `TRUE` (padrao) aplica `fix_coords()` em `lat/lon`.
`nd_policy`	Politica para valores censurados (ND/<LD/<LOQ). Opcoes: `"ld2"` (metade do limite, padrao), `"ld"` (limite), `"zero"` (0), `"na"` (NA_real_).

Value

Um tibble com:

- nomes de colunas normalizados (minusculas, espacos -> _, sem nao-alfanum);
- colunas numericas parseadas ignorando strings de unidade;
- `data` parseada para `Date` (tenta `ymd` e depois `dmy`);
- `ponto` coerido para `character` (quando presente);
- `lat/lon` corrigidos quando `normalize_coords = TRUE`.

Parsed numeric candidates

`c("ph","od","turbidez","dbo","coliformes","p_total","ptotal","fosforo_total","temperatura","ec","co`

Valores censurados (ND/<LD/<LOQ)

O pacote implementa uma politica explicita para tratamento de valores censurados. Valores como `"<0.01"`, `"<LD"`, `"<LOD"`, `"<LOQ"`, `"ND"` sao detectados e tratados conforme a politica especificada em `nd_policy`. O padrao `"ld2"` usa metade do limite de deteccao (recomendacao conservadora).

See Also

`clean_units()`, `validate_wq()`, `conama_check()`, `iqa()`

Examples

```
## Not run:
tmp <- tempfile(fileext = ".csv")
writeLines(
  c("ponto;data;ph;od;turbidez;lat;lon",
    "R1_01;2025-01-20;7,2;6,8;5,1;-163456789;-396543210",
    "R1_01;21/01/2025;7.1;7.0;4.8 mg/L;-16.3456789;-39.6543210"),
  tmp
)
x <- read_wq(tmp)
str(x)

## End(Not run)
```

---

render_report *Render a water-quality report from the internal R Markdown template*

Description

Renders an HTML report using the package's internal R Markdown template. By default, the output is written to a **temporary directory** to comply with CRAN policies. The function returns (invisibly) the full path to the generated file.

Usage

```
render_report(
  df,
  meta = list(river = NA, period = NA),
  output_file = "wq_report.html",
  output_dir = tempdir(),
  template = system.file("templates", "report_rmd.Rmd", package = "tikatuwq")
)
```

Arguments

df	Data frame with the input data used by the template.
meta	Named list with contextual metadata (e.g., river, period).
output_file	File name for the report (default "wq_report.html").
output_dir	Directory where the file will be written (default tempdir()). It will be created if it does not exist.
template	Path to the internal template file. Defaults to the package Rmd template shipped under inst/templates/report_rmd.Rmd.

Details

The template expects a data frame with columns compatible with the package (e.g., ponto, data, parameters used by IQA/CONAMA checks). You can pass optional metadata via meta, such as river and period.

This function relies on **rmarkdown** (listed in Suggests). If the package is not available, an informative error is thrown.

Value

Invisible character string: the absolute path to the generated report.

Notes

- The default output directory is tempdir() to comply with CRAN policies. All files (including intermediate files generated during rendering) are written only to output_dir or temporary directories, never to the package installation directory.
- The template is an **Rmd** (R Markdown). If you prefer Quarto, provide a custom template path to a .qmd and ensure your environment supports it.

See Also

rmarkdown::render()

Examples

```
# Minimal example (writes to a temporary directory)
d <- wq_demo
path <- render_report(d, meta = list(river = "Example River", period = "Jan-Feb"))
file.exists(path)
```

`resume_wq` *Descriptive summaries by group*

Description

Computes basic descriptive statistics (mean, median, sd) for all **numeric** columns in `df`, grouped by one or more keys.

Usage

```
resume_wq(df, by = c("ponto", "mes"), funs = c("mean", "median", "sd"))
```

Arguments

`df`	A data frame or tibble.
`by`	Character vector with grouping column names (default `c("ponto","mes")`). Any names not present in `df` are ignored.
`funs`	**Deprecated** (kept for compatibility; ignored). The function always computes `mean`, `median` and `sd` with `na.rm = TRUE`.

Details

- Grouping columns not found in `df` are silently dropped.
- If no grouping columns remain, an error is thrown.
- Only numeric columns are summarized; if none exist, an error is thrown.
- Missing values are ignored (`na.rm = TRUE`).

Value

A tibble with the grouping keys and one column per statistic/variable, named as `{var}_{stat}` (e.g., `od_mean`, `od_median`, `od_sd`).

See Also

`dplyr::summarise()`, `dplyr::across()`

Examples

```
# Using the demo dataset shipped with the package
d <- wq_demo
# Example: group by point (ponto)
s1 <- resume_wq(d, by = "ponto")
head(s1)

# Example: group by point and month (if 'mes' exists in your data)
# s2 <- resume_wq(d, by = c("ponto", "mes"))
```

`trend_param` *Tendencia monotona por parametro e ponto (Theil-Sen + Spearman)*

Description

Calcula a inclinacao de Theil-Sen (robusta) e o p-valor do teste de correlacao de Spearman entre tempo e o valor do parametro. Retorna estatisticas por grupo (ex.: rio, ponto).

Usage

```
trend_param(
  data,
  param,
  date_col = "data",
  group_cols = c("rio", "ponto"),
  min_n = 6,
  alpha = 0.05
)
```

Arguments

`data`	data.frame com pelo menos uma coluna de data e a coluna do parametro.
`param`	nome do parametro (string), por exemplo "turbidez" ou "iqa".
`date_col`	nome da coluna de datas. Default: "data".
`group_cols`	vetor de nomes para agrupar. Default: c("rio","ponto").
`min_n`	amostra minima por grupo. Default: 6.
`alpha`	nivel de significancia para classificar tendencia. Default: 0.05.

Value

data.frame com colunas por grupo e: n, date_min, date_max, days_span, slope_per_year, intercept, rho_spearman, p_value, trend ("aumento" / "queda" / "estavel"), pct_change_period (aprox. % no periodo observado).

Examples

```
set.seed(1)
df <- data.frame(
  data = as.Date("2024-01-01") + 0:11*30,
  rio = "Demo", ponto = "P1",
  turbidez = 20 + (-0.3)*(0:11) + rnorm(12, 0, 1)
)
trend_param(df, param = "turbidez")
```

`validate_wq` *Validate presence of required columns*

Description

Ensures a minimal set of columns exists in the dataset; otherwise throws an error listing the missing names.

Usage

```
validate_wq(
  df,
  required = c("ph", "turbidez", "od", "dbo", "nt_total", "p_total", "tds",
    "temperatura", "coliformes"),
  nd_policy = c("ld2", "ld", "zero", "na")
)
```

Arguments

`df`	Input data.frame/tibble to validate.
`required`	Character vector of required column names to check for.
`nd_policy`	Policy for censored values (ND/<LD/<LOQ) when required columns are not numeric. One of: • "ld2" (default): use half the detection limit • "ld" : use the detection limit • "zero" : replace with 0 • "na" : replace with NA

Value

The input `df` if valid; otherwise, an error is thrown.

See Also

`read_wq()`, `conama_check()`

Examples

```
df_ex <- data.frame(
  ph = 7, turbidez = 2, od = 7, dbo = 3,
  nt_total = 0.8, p_total = 0.05, tds = 150,
  temperatura = 24, coliformes = 200
)
validate_wq(df_ex)
```

---

wq_demo *Example water quality dataset (subset of real data)*

Description

A small subset of real monitoring data used in examples and vignettes. Now includes extra columns `rio`, `lat`, `lon`.

Real water quality measurements collected by INEMA (Instituto do Meio Ambiente e Recursos Hidricos, Bahia) during monitoring campaigns conducted between 2021 and 2024 in the Rio Buranhem watershed, municipality of Porto Seguro, Bahia, Brazil. Incorporated into the package for demonstration, reproducibility and methodological illustration, following the analytical workflow implemented in tikatuwq. Parameters include sampling dates, site identifiers and multiple physicochemical variables measured during field campaigns.

Usage

```
data(wq_demo)

data(wq_demo)
```

Format

A tibble/data.frame with 20 rows and 14 columns:

rio character, river name

ponto character, monitoring point id

data Date, sampling date

ph numeric, pH

od numeric, dissolved oxygen (mg/L)

turbidez numeric, NTU

dbo numeric, mg/L

coliformes numeric, MPN/100 mL

p_total numeric, total phosphorus (mg/L)

nt_total numeric, total nitrogen (mg/L)

temperatura numeric, degrees Celsius

tds numeric, total dissolved solids (mg/L)

lat numeric, latitude

lon numeric, longitude

A tibble with columns documented in `wq_demo`.

Details

The dataset is a real subset selected from BURANHEM river (dataset-real.csv), used for reproducible examples and vignettes. Covers 4 monitoring points and years 2020–2024.

The dataset has total dissolved solids (`tds`) but **not total solids** (`solidos_totais`), one of the nine CETESB IQA parameters. TDS is not interchangeable with total solids, so `iqa(wq_demo)` returns `NA` and `iqa(wq_demo, allow_partial = TRUE)` gives partial estimates from eight parameters with renormalized weights. These are not CETESB IQA values, and no published IQA exists for these samples: `wq_demo` is example data, not an independent benchmark.

Source

Subset of dataset-real.csv (BURANHEM river, 4 sites, years 2020–2024).

See Also

`iqa()`, `conama_check()`, `plot_series()`, `plot_box()`, `plot_iqa()`, `plot_heatmap()`

`wq_demo`

Examples

```
data("wq_demo", package = "tikatuwq")
head(wq_demo)
# quick IQA example:
# iqa(wq_demo, allow_partial = TRUE)
```

`wq_pca` *Analise de Componentes Principais (PCA) de parametros de qualidade da agua*

Description

Wrapper simplificado sobre `stats::prcomp()` para dados de qualidade da agua. Retorna o objeto PCA, scores por amostra, contribuicao das variaveis (loadings), variancia explicada e dois graficos prontos (biplot e screeplot).

Usage

```
wq_pca(df, params = NULL, color_by = NULL, label_by = NULL, n_components = 4L)
```

Arguments

df	Data frame com os parametros a incluir na analise.
params	Character vector; nomes das colunas a usar. Se NULL (default), todas as colunas numericas sao usadas (excluindo lat, lon, e colunas com sufixo _ok, _status, _delta).
color_by	Character ou NULL; coluna para colorir as amostras no biplot (ex.: "ponto", "season").
label_by	Character ou NULL; coluna para rotular as amostras no biplot (ex.: "ponto").
n_components	Integer; numero de componentes a reter. Default 4.

Details

Apenas colunas numericas sao consideradas. Linhas com qualquer NA nos parametros selecionados sao removidas (com aviso). O PCA e sempre realizado sobre dados centrados e escalonados (scale. = TRUE, center = TRUE).

Os graficos sao retornados como atributos do resultado:

- attr(resultado, "biplot") – dispersao das amostras nos eixos PC1 x PC2, coloridas por color_by.
- attr(resultado, "screeplot") – variancia explicada por componente.
- attr(resultado, "loadings_plot") – contribuicao das variaveis no plano PC1 x PC2.

Value

Uma lista com:

pca Objeto prcomp.

scores Tibble com scores (PC1...PCn) por amostra, mais as colunas de agrupamento (color_by, label_by).

loadings Tibble com loadings das variaveis.

variance Tibble com variancia explicada e acumulada por componente.

Os atributos "biplot", "screeplot" e "loadings_plot" contem objetos ggplot.

See Also

param_analysis()

Examples

```
data("wq_demo", package = "tikatuwq")
res <- wq_pca(wq_demo, color_by = "ponto")
print(res$variance)
attr(res, "biplot")
attr(res, "screeplot")
```

Index

∗ **conama-tools**
  balnear_check, 5
  conama_freq_check, 16
∗ **datasets**
  wq_demo, 58
∗ **load-tools**
  compute_load, 13
  exceedance_prob, 21
∗ **multivariate-tools**
  wq_pca, 59
∗ **parameter-tools**
  param_plot, 38
  param_plot_multi, 39
  param_summary, 40
  param_summary_multi, 41
  param_trend, 42
  param_trend_multi, 43
∗ **reporting-tools**
  generate_analysis, 23
∗ **reporting**
  render_report, 53
∗ **seasonal-tools**
  assign_season, 3
  compare_seasons, 12
∗ **trend-tools**
  mk_seasonal, 34
∗ **visualization-tools**
  plot_iet, 45
  plot_map_quality, 48
∗ **water-quality-indices**
  nsfwqi, 36

assign_season, 3, *13*
assign_season(), *13*, *35*

balnear_check, 5, *17*

classify_iqa, 7, *11*, *30*
classify_tsi_carlson, 8
classify_tsi_lamparelli, 9
clean_units, 9
clean_units(), *53*
compare_iqa, 10
compare_seasons, *4*, 12
compare_seasons(), *4*
compute_load, 13, *22*
conama_check, 15
conama_check(), *6*, *17*, *19*, *23*, *25*, *27*, *53*, *57*, *59*
conama_freq_check, *6*, 16
conama_freq_check(), *6*, *22*
conama_limits, 17
conama_limits(), *15*
conama_report, 18
conama_report(), *15*, *17*, *19, 20*
conama_summary, 19
conama_summary(), *15*, *18*, *20*
conama_text, 20
conama_text(), *15*, *18, 19*

dplyr::across(), *55*
dplyr::summarise(), *55*

exceedance_prob, *14*, 21
exceedance_prob(), *14*

fix_coords, 22

generate_analysis, 23

iet_carlson, 24
iet_carlson(), *27*, *46*, *49*
iet_lamparelli, 26
iet_lamparelli(), *25*, *46*
iqa, *10, 11*, 27
iqa(), *23*, *25*, *27*, *44*, *47*, *49–51*, *53*, *59*
iqa_components, *29*, 31
iqa_methods, *11*, 33

mk_seasonal, 34

normalize_param_names, 36
nsfwqi, 36

param_analysis(), *60*
param_plot, 38, *40–43*
param_plot_multi, *39*, 39, *41–43*
param_summary, *39*, *40*, 40, *42*, *43*
param_summary_multi, *39–41*, 41, *42*, *43*
param_trend, *39–42*, 42, *43*
param_trend_multi, *39–42*, 43
plot_box, 44
plot_box(), *47*, *50*, *59*
plot_heatmap, 44
plot_heatmap(), *44*, *50*, *59*
plot_iet, 45, *49*
plot_iqa, 46
plot_iqa(), *46*, *59*
plot_map, 47
plot_map(), *49*
plot_map_quality, *46*, 48
plot_series, 50
plot_series(), *44*, *47*, *51*, *59*
plot_trend, 50

read_wq, 52
read_wq(), *10*, *57*
render_report, 53
resume_wq, 55

trend_param, 56
trend_param(), *35*

validate_wq, 57
validate_wq(), *53*

wq_demo, 58, *59*
wq_pca, 59