Package 'tikatuwq'

November 17, 2025

Type Package

Title Water Quality Assessment and Environmental Compliance in Brazil

Version 0.8.0

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), the Trophic State Index (TSI/IET) after Carlson (1977) <doi:10.4319/lo.1977.22.2.0361> and Lamparelli (2004) <https://www.teses.usp.br/teses/disponiveis/41/41134/tde-20032006-075813/publico/TeseLamparelli2004.pdf>, and the National Sanitation Foundation Water Quality Index (NSF WQI) <doi:10.1007/s11157-023-09650-7>. 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> and generates reproducible reports for routine monitoring workflows.
The example dataset (`wq_demo`) is now a real subset from monitoring data (BURANHEM river, 2020-2024, 4 points, 20 rows, 14 columns including extra `rio`, `lat`, `lon`). All core examples and vignettes use this realistic sample, improving reproducibility and documentation value for users.

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, leaflet

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

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>)

Repository CRAN

Date/Publication 2025-11-17 21:50:39 UTC

Contents

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

Description

Converte valores numericos de IQA (0-100) em classes qualitativas padronizadas. Suporta rotulos em portugues ("pt") ou ingles ("en").

Usage

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

Arguments

x	Vetor numerico com IQA em 0-100. Valores NA sao preservados.
locale	Idioma dos rotulos: "pt" (padrao) ou "en".

Value

Um fator ordenado com os rotulos de classe.

Examples

```
classify_iqa(c(15, 40, 65, 80, 95))
classify_iqa(c(15, 40, 65, 80, 95), locale = "en")
```

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)
```

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).

If multiple limit rows exist for the same parameter, `*_ok` is TRUE if any row is satisfied; for `status/lim_min/lim_max/delta`, the first satisfied row is chosen; if none satisfy, the row with the smallest absolute violation (min |delta|) is used.

Usage

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

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").

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_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)
```

`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):

- `TSI_Secchi = 60 - 14.41 * log10(secchi)`
- `TSI_Chla = 9.81 * log10(clorofila) + 30.6`
- `TSI_TP = 14.42 * log10(tp) + 4.15`

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

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

Implementacao pragmatica; confirme coeficientes/limiares para seu contexto regulatorio. 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 de Secchi (0-100).
- `IET_Lamp` — indice Lamparelli agregado (0-100).
- `TSI_status` — classe qualitativa (quando `add_status=TRUE`).
- `ambiente` — tipo de ambiente informado.

See Also

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

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

Description

Computa o IQA/WQI combinando subindices (Qi) por **media ponderada**. Os subindices sao obtidos por interpolacao linear por trechos sobre curvas aproximadas (estilo CETESB/NSF).

Usage

```
iqa(
  df,
  pesos = c(od = 0.17, coliformes = 0.15, dbo = 0.1, nt_total = 0.1, p_total = 0.1,
    turbidez = 0.08, tds = 0.08, pH = 0.12, temperatura = 0.1),
  method = c("CETESB_approx"),
  na_rm = FALSE,
  add_status = TRUE,
  locale = c("pt", "en"),
  ...
)
```

Arguments

`df`	Data frame (ou tibble) com as colunas requeridas. Nomes esperados (portugues): `od`, `coliformes`, `dbo`, `nt_total`, `p_total`, `turbidez`, `tds`, `ph` (ou `pH`), `temperatura`.
`pesos`	Pesos nomeados para cada parametro. Padroes seguem pratica CETESB/NSF: `od=.17`, `coliformes=.15`, `dbo=.10`, `nt_total=.10`, `p_total=.10`, `turbidez=.08`, `tds=.08`, `pH=.12`, `temperatura=.10`.
`method`	Conjunto de curvas de interpolacao; atualmente apenas `"CETESB_approx"`.

na_rm	Logico; se FALSE (padrao), linhas com Qi ausentes geram erro. Se TRUE, o IQA e computado usando apenas os parametros disponiveis e o denominador e ajustado para a soma dos pesos presentes.
add_status	Logico; se TRUE (padrao), adiciona a coluna IQA_status com a classificacao qualitativa (0-100).
locale	Idioma de IQA_status: "pt" (padrao) ou "en".
...	Reservado para uso futuro (ignorado).

Details

Compatibilidade de nomes:

- A tabela de curvas usa a chave "pH". Se seus dados possuem ph (minusculo), a curva "pH" e mapeada para a coluna ph.
- Para temperatura, a coluna temp (alias comum) e automaticamente aceita caso temperatura nao exista.

Se as curvas internas retornarem Qi em 0-10 (variante historica), o valor agregado e normalizado internamente para 0-100 antes do retorno. Valores finais sao limitados ao intervalo [0, 100].

Value

O df de entrada com a coluna numerica IQA (0-100) e, quando add_status = TRUE, a coluna fator IQA_status. O atributo "iqa_method" e definido no objeto retornado.

Examples

```
d <- wq_demo
d2 <- iqa(d, na_rm = TRUE)
table(d2$IQA_status, useNA = "ifany")
```

---

nsfwqi *NSF Water Quality Index (NSF WQI, prototype)*

Description

Computes a **prototype** NSF WQI as a weighted arithmetic mean of parameter sub-scores (Qi) using simple piecewise rules. This is intended for quick demonstrations and is **not** a full replication of the original NSF curves.

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, sst = 0.07),
  na_rm = FALSE
)
```

Arguments

df	Data frame containing columns compatible with the mapping above.
pesos	Named numeric vector with parameter weights. Defaults follow a common NSF WQI variant: do=.17, fc=.16, ph=.11, bod=.11, temp_change=.10, po4=.10, no3=.10, turbidez=.08, sst=.07.
na_rm	Logical; allow NA per row and rescale weights to available parameters (TRUE) or error on missing inputs (FALSE).

Details

The function accepts both NSF-style column names and common Brazilian aliases. The mapping tried (if present) is:

- do <- od
- fc <- coliformes
- ph <- pH or ph
- bod <- dbo
- turbidez stays turbidez
- sst <- solidos_suspensos
- po4 <- po4 or p_ortofosfato
- no3 <- no3 or n_nitrato
- temp_change must be supplied as-is (delta T to reference)

If na_rm = TRUE, weights are rescaled **per row** to the parameters available in that row. If na_rm = FALSE (default), any missing required input leads to an error.

Value

The input df with an added numeric column NSFWQI.

Examples

```
d <- wq_demo
# create minimal aliases so the prototype can run
d$do  <- d$od
d$fc  <- d$coliformes
d$ph  <- d$ph
d$bod <- d$dbo
# others are missing; use na_rm = TRUE to rescale weights by row
out <- nsfwqi(d, na_rm = TRUE)
head(out$NSFWQI)
```

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_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, na_rm = 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

```
## Not run:
d  <- read_wq("dataset-real.csv")
d2 <- iqa(d, na_rm = 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)")

## End(Not run)
```

`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 avoid writing into the user's project folder during examples or automated checks.
- 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`.

This dataset contains 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. The dataset was 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/data.frame. See `wq_demo` documentation for column details.

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. All core columns for IQA/CONAMA/plotting helpers are present.

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, na_rm = TRUE)
```

Index