Package 'tejoR'

August 6, 2026

Title Statistical Harmonization of Territorial Series Across Changing Geographies

Version 0.2.2

Description Builds, validates, seals and applies weighting matrices ('crosswalks') to carry statistical series across changing zoning systems, such as the transition from the 112 Unidades de Planeamiento Zonal (UPZ) to the 33 Unidades de Planeamiento Local (UPL) in Bogota (Decree 555 of 2021). Implements the 'tejo-crosswalk/0.2' specification shared with the 'Python' package 'tejo': 'sha256'-sealed artifacts, non-negative weights that sum to one for each source unit, dasymetric weighting with vector ancillary data via 'sf', rates that are never interpolated directly, and missing values that propagate instead of being silently imputed. Methods: Tobler (1979) <doi:10.1080/01621459.1979.10481647>; Mennis (2003) <doi:10.1111/0033-0124.10042>. Descripcion en espanol: construye, valida, sella y aplica matrices de ponderadores ('crosswalks') para trasladar series estadisticas entre mallas geograficas que cambian, como la transicion de UPZ a UPL en Bogota (Decreto 555 de 2021): integridad por 'sha256', pesos no negativos que suman uno por unidad fuente, metodo dasimetrico con ancilar vectorial via 'sf', tasas que nunca se interpolan directamente y valores faltantes que se propagan en lugar de imputarse.

License MIT + file LICENSE

URL https://github.com/adriGr52/tejoR

BugReports https://github.com/adriGr52/tejoR/issues

Encoding UTF-8

Depends R (>= 4.1.0)

Imports sf, jsonlite, digest, stats, utils

Suggests testthat

RoxygenNote 7.3.1

NeedsCompilation no

Author Luz Adriana Gutierrez Rodriguez [aut, cre]

Maintainer Luz Adriana Gutierrez Rodriguez <adrig63@gmail.com>

Repository CRAN

Date/Publication 2026-08-06 13:50:02 UTC

Contents

`aplicar_crosswalk` *Aplica un crosswalk a datos tabulares segun el tipo de variable*

Description

Reglas duras compartidas con el gemelo 'Python': las variables EXTENSIVAS se suman con W; las INTENSIVAS son media ponderada por la masa declarada (o por el area de interseccion si no se declara, supuesto documentado); las TASAS jamas se interpolan directamente: se declaran como `c(nombre, numerador, denominador)`, se trasladan ambos componentes y el cociente devuelve `NA` (nunca `Inf`) ante denominador cero. Los `NA` contaminan a sus destinos: sin imputacion silenciosa.

Usage

```
aplicar_crosswalk(
  cw,
  datos,
  extensivas = character(),
  intensivas = character(),
  tasas = list(),
  masa = NULL
)
```

Arguments

`cw`	Objeto `tejo_crosswalk`.
`datos`	data.frame con una fila por unidad fuente (debe traer la columna identificadora de la fuente).
`extensivas, intensivas`	Vectores con nombres de columnas.
`tasas`	Lista de vectores `c(nombre_salida, col_numerador,col_denominador)`.
`masa`	Nombre de la columna de masa para las intensivas (recomendado, p. ej. poblacion).

Value

data.frame con una fila por unidad destino y las variables trasladadas; para cada tasa agrega <nombre>_num y <nombre>_den. Incluye los atributos fuentes_sin_datos y fuentes_fuera_del_crosswalk.

Examples

```
rect <- function(x0, y0, x1, y1)
  sf::st_polygon(list(rbind(c(x0, y0), c(x1, y0), c(x1, y1),
                            c(x0, y1), c(x0, y0))))
src <- sf::st_sf(sid = "S1", geometry = sf::st_sfc(rect(0, 0, 2, 2)),
                 crs = 3116)
tgt <- sf::st_sf(tid = c("T1", "T2"),
                 geometry = sf::st_sfc(rect(0, 0, 1, 2), rect(1, 0, 2, 2)),
                 crs = 3116)
cw <- construir_crosswalk(src, tgt, "sid", "tid", method = "area")
datos <- data.frame(sid = "S1", casos = 20, pob = 400)
aplicar_crosswalk(cw, datos, extensivas = "casos",
                  tasas = list(c("tasa", "casos", "pob")))
```

---

cargar_crosswalk *Guardar y cargar crosswalks sellados*

Description

guardar_crosswalk escribe <base>.csv mas <base>.meta.json con sello 'SHA-256' de la tabla y numericos a 17 digitos significativos (fidelidad de doble precision). cargar_crosswalk lee el par de archivos y, con validate = TRUE, verifica version de la especificacion, integridad del sello, columnas minimas e invariantes. Un artefacto alterado en un solo byte produce error de integridad.

Usage

```
cargar_crosswalk(base, validate = TRUE)

guardar_crosswalk(cw, base, extra_meta = list())
```

Arguments

base	Ruta base de los archivos, sin extension (se agregan .csv y .meta.json).
validate	Logico; si TRUE (defecto) verifica sello e invariantes al cargar.
cw	Objeto tejo_crosswalk a sellar.
extra_meta	Lista opcional de metadatos adicionales a incorporar.

Value

guardar_crosswalk devuelve, de forma invisible, la lista de metadatos sellados (incluye sha256_tabla y spec_version). cargar_crosswalk devuelve el objeto tejo_crosswalk reconstruido (lista con tabla y meta).

Examples

```
rect <- function(x0, y0, x1, y1)
  sf::st_polygon(list(rbind(c(x0, y0), c(x1, y0), c(x1, y1),
                            c(x0, y1), c(x0, y0))))
src <- sf::st_sf(sid = "S1", geometry = sf::st_sfc(rect(0, 0, 2, 2)),
                 crs = 3116)
tgt <- sf::st_sf(tid = c("T1", "T2"),
                 geometry = sf::st_sfc(rect(0, 0, 1, 2), rect(1, 0, 2, 2)),
                 crs = 3116)
cw <- construir_crosswalk(src, tgt, "sid", "tid", method = "area")
base <- tempfile("cw_")
guardar_crosswalk(cw, base, extra_meta = list(nota = "ejemplo"))
cw2 <- cargar_crosswalk(base, validate = TRUE)
all.equal(cw2$tabla$weight, cw$tabla$weight)
```

`construir_crosswalk` *Construye la matriz de ponderadores W entre dos mallas*

Description

Estima w_ij, la fraccion de cada unidad fuente asignada a cada unidad destino, por ponderacion de area o dasimetrica (masa continua de una capa ancilar, con reparto 'areal' o por punto interior 'centroid'). Garantias: sin reproyeccion silenciosa (CRS proyectado identico obligatorio), fuentes con masa ancilar nula caen a pesos de area de forma marcada (columna `fallback_area`) y los invariantes se verifican al construir.

Usage

```
construir_crosswalk(
  source,
  target,
  source_id,
  target_id,
  method = c("area", "dasymetric"),
  ancillary = NULL,
  ancillary_mass = NULL,
  ancillary_assignment = c("areal", "centroid"),
  renormalize = TRUE,
  coverage_warn = 0.99
)
```

Arguments

`source`, `target`	Objetos `sf` poligonales en el mismo CRS proyectado.
`source_id`, `target_id`	Nombres de las columnas identificadoras.
`method`	`"area"` o `"dasymetric"`.

ancillary	Capa sf con la masa ancilar (p. ej., manzanas censales con poblacion). Obligatoria para el metodo dasimetrico.
ancillary_mass	Nombre de la columna de masa en ancillary.
ancillary_assignment	"areal" (masa repartida proporcional al area del poligono ancilar en cada pieza) o "centroid" (toda la masa al punto interior).
renormalize	Logico; renormaliza por cobertura para tolerar slivers de borde, dejando registro de la cobertura bruta por fuente.
coverage_warn	Umbral de cobertura bajo el cual se agrega una advertencia a los metadatos.

Value

Objeto tejo_crosswalk: lista con tabla (data.frame con weight, w_area, area_interseccion, cobertura_area, fallback_area y, si aplica, masa_int) y meta (metodo, CRS, conteos, cobertura, advertencias).

Examples

```
rect <- function(x0, y0, x1, y1)
  sf::st_polygon(list(rbind(c(x0, y0), c(x1, y0), c(x1, y1),
                            c(x0, y1), c(x0, y0))))
src <- sf::st_sf(sid = "S1", geometry = sf::st_sfc(rect(0, 0, 2, 2)),
                 crs = 3116)
tgt <- sf::st_sf(tid = c("T1", "T2"),
                 geometry = sf::st_sfc(rect(0, 0, 1, 2), rect(1, 0, 2, 2)),
                 crs = 3116)
mz <- sf::st_sf(pob = c(1, 1, 9, 9),
                geometry = sf::st_sfc(rect(0, 0, 1, 1), rect(0, 1, 1, 2),
                                      rect(1, 0, 2, 1), rect(1, 1, 2, 2)),
                crs = 3116)
cw <- construir_crosswalk(src, tgt, "sid", "tid", method = "dasymetric",
                          ancillary = mz, ancillary_mass = "pob")
cw$tabla[, c("sid", "tid", "weight")]  # 0.1 y 0.9: manda la masa, no el area
```

---

validar_crosswalk *Valida los invariantes de un crosswalk*

Description

Re-verifica el contrato de la especificacion 'tejo-crosswalk/0.2': pesos no negativos y filas que suman 1 por unidad fuente (pycnofilaxis), salvo que los metadatos declaren renormalize = FALSE.

Usage

```
validar_crosswalk(cw, atol = 1e-06)
```

Arguments

cw	Objeto de clase tejo_crosswalk (lista con tabla y meta), creado por construir_crosswalk o cargar_crosswalk.
atol	Tolerancia absoluta para la suma de pesos por fuente.

Value

Lista con el reporte de invariantes: n_fuentes, n_pares, no_negatividad, filas_suman_1 y desviacion_max_suma_filas. Detiene con error si se violan.

Examples

```
rect <- function(x0, y0, x1, y1)
  sf::st_polygon(list(rbind(c(x0, y0), c(x1, y0), c(x1, y1),
                            c(x0, y1), c(x0, y0))))
src <- sf::st_sf(sid = "S1", geometry = sf::st_sfc(rect(0, 0, 2, 2)),
                 crs = 3116)
tgt <- sf::st_sf(tid = c("T1", "T2"),
                 geometry = sf::st_sfc(rect(0, 0, 1, 2), rect(1, 0, 2, 2)),
                 crs = 3116)
cw <- construir_crosswalk(src, tgt, "sid", "tid", method = "area")
validar_crosswalk(cw)
```

Index