Package 'tissot'

July 12, 2026

Title Tissot Indicatrix for Map Projection Distortion

Version 0.3.0

Description Compute and visualize the 'Tissot Indicatrix' for map projections. The indicatrix characterizes projection distortion by computing scale factors, angular deformation, areal distortion, and convergence at arbitrary points. Based on the calculations shared by Bill Huber on <https://gis.stackexchange.com/a/5075/482>. Uses 'PROJ' for coordinate transformation and distortion factor computation. Developed using the method published in Snyder, JP (1987) <doi:10.3133/pp1395>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 3.6.0)

Imports tibble, PROJ, graphics, grDevices, stats

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

Config/testthat/edition 3

URL https://hypertidy.github.io/tissot/, https://github.com/hypertidy/tissot

BugReports https://github.com/hypertidy/tissot/issues

Language en-US

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Michael Sumner [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-2471-7511>), Bill Huber [aut] (original algorithm and calculations)

Maintainer Michael Sumner <mdsumner@gmail.com>

Repository CRAN

Date/Publication 2026-07-12 15:50:02 UTC

Contents

`indicatrix` *Build Tissot indicatrix objects*

Description

Computes an `indicatrix_list` containing per-point distortion objects suitable for plotting with `plot.indicatrix_list()` or querying directly.

Usage

```
indicatrix(x, target = NULL, ..., source = "EPSG:4326")
```

Arguments

`x`	a `tissot_tbl`, or any xy-ish input (see `tissot()`)
`target`	target projection CRS (extracted from `tissot_tbl` attributes if `x` is one; required otherwise)
`...`	passed to `tissot()`
`source`	source CRS (default `"EPSG:4326"`)

Details

`indicatrix()` accepts either:

- A `tissot_tbl` object (from `tissot()`) — projection is extracted from attributes, `target` is optional
- Any xy-ish input with an explicit `target`

Value

An `indicatrix_list` object (a list of `indicatrix` objects with `source` and `target` stored as attributes)

See Also

tissot(), plot.indicatrix(), plot.indicatrix_list(), ti_ellipse()

Examples

```
## From a tissot_tbl
r <- tissot(cbind(seq(-150, 150, by = 30), 0), "+proj=robin")
ii <- indicatrix(r)
plot(ii, scale = 5e5)

## From raw coordinates
ii2 <- indicatrix(c(0, 45), "+proj=stere +lat_0=90")
plot(ii2)
```

---

plot.indicatrix *Plot an indicatrix*

Description

Draws a single Tissot indicatrix ellipse on the current plot. The ellipse shows the distortion of a unit circle under the map projection. Optional overlays include a reference unit circle, and lambda/phi direction axes.

Usage

```
## S3 method for class 'indicatrix'
plot(
  x,
  scale = 1e+05,
  n = 72,
  col = "#FF990055",
  border = "black",
  add = TRUE,
  show.axes = TRUE,
  show.circle = TRUE,
  ...
)
```

Arguments

x	an indicatrix object (from indicatrix())
scale	scaling factor for the ellipse size in projected units
n	number of points on the ellipse
col	fill colour for the ellipse
border	border colour
add	logical; add to existing plot (default TRUE)?

show.axes	TRUE, FALSE, or a named list of graphical parameters for the direction lines. Defaults: list(col.lambda = "red", col.phi = "blue", lwd = 1.5).
show.circle	TRUE, FALSE, or a named list of graphical parameters for the reference circle. Defaults: list(col = adjustcolor("white", alpha.f = 1), border = "grey70", lwd = 4.5, lty = 2).
...	passed to graphics::polygon()

Details

show.circle and show.axes accept TRUE (use defaults), FALSE (hide), or a named list of graphical parameters to override defaults. For example, show.circle = list(border = "blue", lty = 3).

Value

The input x, invisibly.

See Also

indicatrix(), plot.indicatrix_list(), ti_ellipse()

---

plot.indicatrix_list *Plot a list of indicatrixes*

Description

Draws all indicatrixes in an indicatrix_list, optionally creating a new plot or adding to an existing one. Can colour-code the fill by a distortion metric.

Usage

```
## S3 method for class 'indicatrix_list'
plot(
  x,
  scale = 1e+05,
  n = 72,
  col = "#FF990055",
  border = "black",
  add = FALSE,
  show.axes = TRUE,
  show.circle = TRUE,
  fill.by = NULL,
  palette = NULL,
  ncolors = 64L,
  ...
)
```

Arguments

x	an indicatrix_list (from indicatrix())
scale	scaling factor for ellipse size in projected units
n	number of points per ellipse
col	fill colour. If a single colour, used for all ellipses. If NULL and fill.by is set, colours are generated automatically.
border	border colour
add	logical; add to existing plot? If FALSE, creates a new plot sized to contain all ellipses.
show.axes	TRUE, FALSE, or a named list of graphical parameters. See plot.indicatrix() for defaults.
show.circle	TRUE, FALSE, or a named list of graphical parameters. Default TRUE for the list method (the circle-vs-ellipse comparison makes distortion visible at map scale). See plot.indicatrix() for defaults.
fill.by	character; name of a distortion metric to colour-code the fill. One of "scale_area", "angle_deformation", "scale_h", "scale_k", "scale_a", "scale_b". Default NULL (uniform fill).
palette	colour palette function (default grDevices::hcl.colors())
ncolors	number of colours in the palette (default 64)
...	passed to plot.indicatrix()

Value

The input x, invisibly.

See Also

indicatrix(), plot.indicatrix(), tissot_map()

Examples

```
xy <- expand.grid(seq(-150, 150, by = 30), seq(-60, 60, by = 30))
r <- tissot(xy, "+proj=robin")
ii <- indicatrix(r)

## Uniform fill
plot(ii, scale = 6e5, add = FALSE)
tissot_map()

## Colour by areal distortion
plot(ii, scale = 6e5, add = FALSE, fill.by = "scale_area")
tissot_map()
```

`tissot` *Compute Tissot indicatrix properties*

Description

Compute the Tissot indicatrix at given longitude/latitude locations for a map projection. Returns scale factors, angular deformation, convergence, and related distortion properties.

Usage

```
tissot(
  x,
  target,
  ...,
  source = "EPSG:4326",
  method = c("proj", "finitediff"),
  A = 6378137,
  f.inv = 298.257223563,
  dx = 1e-04
)
```

Arguments

`x`	input coordinates — any xy-ish object: a two-column matrix, data.frame, tibble, list with `x`/`y` or `lon`/`lat` components, or a length-2 numeric vector for a single point
`target`	target projection CRS string (required)
`...`	ignored
`source`	source CRS (default `"EPSG:4326"`)
`method`	computation method: `"proj"` (default) uses `PROJ::proj_factors()`; `"finitediff"` uses a finite-difference Jacobian (Snyder 1987)
`A`	semi-major axis of the ellipsoid (default WGS84; `method = "finitediff"` only)
`f.inv`	inverse flattening (default WGS84; `method = "finitediff"` only)
`dx`	finite difference step in degrees (default 1e-4; `method = "finitediff"` only)

Details

By default (`method = "proj"`) distortion measures are computed via `PROJ::proj_factors()`, which calls the PROJ C library directly and is more accurate than finite differences (particularly for tabular or piecewise-defined projections). The original finite-difference path based on Snyder (1987) — inspired by Bill Huber's formulation at https://gis.stackexchange.com/a/5075/482 — is preserved as `method = "finitediff"`.

`x` is assumed to contain longitude/latitude values; the default source CRS is `"EPSG:4326"`. Set `source` for a different geographic CRS.

Value

A `tissot_tbl` tibble with columns: `x` (lon), `y` (lat), `dx_dlam`, `dy_dlam`, `dx_dphi`, `dy_dphi`, `scale_h`, `scale_k`, `scale_omega`, `scale_a`, `scale_b`, `scale_area`, `angle_deformation`, `convergence`. The `source` and `target` CRS strings are stored as attributes.

See Also

`indicatrix()`, `tissot_map()`, `tissot_unproject()`, `PROJ::proj_factors()`, `PROJ::proj_trans()`

Examples

```
tissot(c(0, 45), "+proj=robin")
tissot(cbind(seq(-180, 180, by = 30), 0), "+proj=robin")

## compare methods
tissot(c(0, 45), "+proj=robin", method = "proj")
tissot(c(0, 45), "+proj=robin", method = "finitediff")
```

`tissot_get_proj` *Get or set the current plot projection*

Description

Deprecated. Prefer passing `target` explicitly.

These functions access the package-level projection state. Prefer passing `target` explicitly to `tissot_map()` and `tissot_abline()`, or let `plot.indicatrix_list()` or `image.tissot_raster()` set it automatically.

Usage

```
tissot_get_proj()

tissot_set_proj(target)
```

Arguments

`target`	projection CRS string

Value

`tissot_get_proj()` returns the current projection string or `NULL`

Examples

```
tissot_set_proj("+proj=robin")
tissot_get_proj()
tissot_set_proj(NULL)  # reset
```

`tissot_map` *Plot world coastline on a projected map*

Description

`tissot_map()` draws the bundled world coastline, projected if a projection is current. The projection is determined in this order:

1. An explicit `target` argument
2. The projection recorded by the most recent call to `plot.indicatrix_list()`, `image.tissot_raster()`, or `tissot_raster()`

Usage

```
tissot_map(..., target = NULL, add = TRUE)

tissot_abline(x, y = NULL, ..., source = "EPSG:4326", target = NULL)
```

Arguments

`...`	graphical parameters passed to `graphics::lines()` (if adding) or `graphics::plot()` (if creating new)
`target`	target CRS. If `NULL`, uses the last plot projection (from `plot.indicatrix_list()`) or draws in lon/lat.
`add`	logical; add to existing plot (default `TRUE`) or create new
`x`	longitude values (or any xy-ish input; see `tissot()`)
`y`	latitude values (ignored if `x` is a matrix)
`source`	source CRS for the coordinates (default `"EPSG:4326"`)

Details

`tissot_abline()` draws vertical and horizontal reference lines at a given longitude/latitude in projected coordinates.

Value

`tissot_map()` invisibly returns the (projected) world coastline matrix

`tissot_abline()` is called for its side effect

See Also

`plot.indicatrix_list()`, `tissot()`

Examples

```
r <- tissot(cbind(seq(-150, 150, by = 30), 0), "+proj=robin")
ii <- indicatrix(r)
plot(ii, scale = 6e5, add = FALSE)
tissot_map()
```

`tissot_raster` *Compute distortion surfaces on a regular grid*

Description

Create a raster-like grid of Tissot distortion metrics in the target projection. The grid is laid out in projected coordinates, back-projected to longitude/latitude, and passed through `tissot()`. Returns a list of matrices compatible with `image()` and easily converted to raster objects (e.g. via `terra::rast()`).

Usage

```
tissot_raster(
  target,
  extent = NULL,
  radius = 2e+07,
  nx = 100L,
  ny = NULL,
  metrics = c("scale_area", "angle_deformation", "scale_h", "scale_k"),
  ...,
  source = "EPSG:4326"
)

## S3 method for class 'tissot_raster'
image(x, metric = NULL, col = NULL, asp = 1, main = NULL, ...)

## S3 method for class 'tissot_raster'
plot(x, ...)
```

Arguments

`target`	target projection CRS string
`extent`	numeric length 4: `c(xmin, xmax, ymin, ymax)` in projected coordinates. If `NULL` (default), estimated from the global lonlat bounding box (clamped by `radius`).
`radius`	maximum half-width of the grid extent in projected units (default `2e7`). The auto-detected extent is clamped so that no dimension exceeds `2 * radius`. Prevents blow-up for projections like stereographic or gnomonic that map the whole globe to huge coordinates. Ignored if `extent` is supplied.
`nx`	integer; number of grid cells in x (default 100)

ny	integer; number of grid cells in y. If NULL, set to maintain approximately square cells.
metrics	character vector of column names from tissot() output to include as layers. Default includes areal scale, angular deformation, and meridional/parallel scale factors.
...	passed to image()
source	source CRS (default "EPSG:4326")
x	a tissot_raster (from tissot_raster())
metric	which metric to plot (default: first available)
col	colour palette (default: hcl.colors(64, "YlOrRd", rev = TRUE))
asp	aspect ratio (default 1)
main	plot title (default: metric name)

Details

If extent is NULL, the function samples a dense set of boundary coordinates spanning $[-180, 180] \times [-85, 85]$ (longitude/latitude), projects them into the target CRS, and uses the resulting bounding range (padded by 5%) as the grid extent.

Value

A tissot_raster list with components:

x numeric vector of x (easting) cell centers

y numeric vector of y (northing) cell centers

z named list of ny x nx matrices, one per metric

target the target CRS string

extent the c(xmin, xmax, ymin, ymax) used

The object has an image() method for quick plotting.

See Also

tissot(), plot.tissot_raster(), image.tissot_raster()

Examples

```
tr <- tissot_raster("+proj=robin", nx = 60)
image(tr)

## specific metric
image(tr, metric = "angle_deformation")

## with coastline overlay
image(tr)
tissot_map()
```

```
## polar stereo — use radius to cap extent
tp <- tissot_raster("+proj=stere +lat_0=-90", radius = 5e6, nx = 60)
image(tp)
tissot_map()
```

`tissot_unproject` *Unproject coordinates to geographic (lon/lat) CRS*

Description

A convenience wrapper around `PROJ::proj_trans()` for converting projected coordinates to geographic. Useful for generating regular grids in a projected CRS to feed to `tissot()`, which requires lon/lat input.

Usage

```
tissot_unproject(x, target = "EPSG:4326", ..., source = NULL)
```

Arguments

`x`	input coordinates — any xy-ish object: a two-column matrix, data.frame, tibble, list with `x`/`y` or `lon`/`lat` components, or a length-2 numeric vector for a single point
`target`	target CRS string (default `"EPSG:4326"`). Must be geographic.
`...`	ignored
`source`	source CRS string (required). Must be a projected CRS.

Value

A two-column matrix of longitude and latitude values.

See Also

`tissot()`, `PROJ::proj_trans()`

Examples

```
## regular grid in UTM zone 55S, unprojected to lon/lat for tissot()
xy <- expand.grid(x = seq(4e5, 6e5, length.out = 5),
                  y = seq(5200000, 5400000, length.out = 4))
ll <- tissot_unproject(xy, source = "EPSG:32755")
tissot(ll, "+proj=utm +zone=55 +south")
```

`ti_ellipse` *Generate ellipse coordinates for an indicatrix*

Description

Generate ellipse coordinates for an indicatrix

Usage

```
ti_ellipse(x, scale = 1e+05, n = 72, ...)
```

Arguments

`x`	an indicatrix object
`scale`	scaling factor for the ellipse size in projected units
`n`	number of points on the ellipse
`...`	ignored

Value

A two-column matrix of projected coordinates tracing the ellipse

See Also

`indicatrix()`, `plot.indicatrix()`

Examples

```
ii <- indicatrix(c(0, 45), "+proj=robin")
ell <- ti_ellipse(ii[[1]], scale = 5e5)
head(ell)

## draw on a map
tissot_map(target = "+proj=robin", add = FALSE)
lines(ell)
```

world *World coastline*

Description

A matrix of longitude/latitude coordinates representing a simplified world coastline, derived from the maps package. Continent boundaries are separated by `NA` rows. Longitudes are constrained to `abs(lon) <= 180`.

Usage

```
world
```

Format

A two-column numeric matrix (longitude, latitude)

[.indicatrix_list *Subset an indicatrix_list*

Description

Subset an indicatrix_list

Usage

```
## S3 method for class 'indicatrix_list'
x[i]
```

Arguments

`x`	an `indicatrix_list`
`i`	indices

Value

An `indicatrix_list` with the selected elements

Index