Package: topocast (via r-universe)

October 14, 2026

Title Moving-Window Regression Downscaling of Raster Data

Version 0.0.5

Description Downscales coarse-resolution raster data to a finer grid by fitting local linear regressions of a response, such as a climate variable, on one or more fine-resolution predictors, such as elevation and other terrain indices, within a moving window. Regression coefficients are estimated for every cell using summed-area tables, so the cost is independent of the window size, then resampled to the target resolution and applied to the fine-resolution predictors. Multiplicative and additive anomaly application downscale time series relative to a baseline climatology, following the regression-on-elevation approach used for high-resolution climate surfaces described in Karger et al. (2017) <doi:10.1038/sdata.2017.122>.

License MIT + file LICENSE

Language en-US

Encoding UTF-8

Depends R (>= 4.1.0)

Imports Rcpp (>= 1.0.0), terra (>= 1.7.0)

Suggests knitr, raster, rmarkdown, sf, stars, svglite, testthat (>= 3.0.0)

LinkingTo Rcpp, RcppArmadillo

SystemRequirements C++17

VignetteBuilder knitr

URL https://gillescolling.com/topocast/, https://github.com/gcol33/topocast

BugReports https://github.com/gcol33/topocast/issues

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

NeedsCompilation yes

Author Gilles Colling [aut, cre, cph] (ORCID:
<https://orcid.org/0000-0003-3070-6066>)

Maintainer Gilles Colling <gilles.colling051@gmail.com>

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

Date/Publication 2026-08-07 17:20:08 UTC

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

RemoteRef HEAD

RemoteSha 5511e785c6d7c47beec3872241e961c524b9317b

Contents

topocast 2
window_regression 6

Index **8**

`topocast` *Downscale a coarse raster onto fine terrain by moving-window regression*

Description

Fits a local linear regression of a coarse response on one or more coarse predictors over a moving window, resamples the resulting intercept and slope grids to the fine grid, and evaluates them on the fine-resolution predictors. The output carries the fine-scale structure of the predictors with locally varying coefficients.

Usage

```
topocast(
  formula,
  data,
  onto,
  radius,
  aggregate = "average",
  coefficients = FALSE,
  diagnostics = FALSE,
  anomaly = NULL,
  baseline = NULL,
  type = c("ratio", "additive"),
  method = NULL,
  output = NULL,
  clamp = FALSE,
  min_cells = 0L,
  min_variance = 1e-08,
  threads = NULL
)
```

Arguments

formula	A two-sided formula of bare layer names, such as `prec ~ elev + slope`. The left-hand side names the coarse response layer in `data`, or `cbind(prec, tmin)` for several responses sharing the predictors; the right-hand side names the predictor layers.
data	A gridded coarse input holding the response layer and, optionally, predictor layers named in `formula`: a `SpatRaster`, a `Raster*` (raster), or a `stars` object. Any predictor not in `data` is derived from `onto`.
onto	The target. A gridded `SpatRaster`, `Raster*`, or `stars` object whose grid defines the output, holding every predictor layer named in `formula`; or an `sf`/`SpatVector` of points carrying those predictors as attributes, in which case the fit is evaluated at the points. With a point `onto` every predictor must be a layer of `data` (the derive-from-`onto` shortcut needs a grid).
radius	Integer window radius in coarse cells, at least `1`; the window is a square of side `2 * radius + 1`. A radius of `0` is rejected: its one-cell window can never hold the valid cells a fit needs.
aggregate	Resampling method used to derive a coarse predictor from `onto` when it is not already a layer of `data`, passed to `terra::resample()`. Default `"average"`.
coefficients	If `TRUE`, return the fitted layer together with the `(Intercept)` and per-predictor slope grids on the `onto` grid. Not supported with `anomaly`. Default `FALSE`. With `output = "raster"`, the `raster` package's own `names()` accessor rewrites `"(Intercept)"` to `"X.Intercept."` (via `make.names()`, unconditionally, on every read); every other output class keeps the name verbatim.
diagnostics	If `TRUE`, also return three grids (or columns) describing the local fit, each brought onto `onto`: `r.squared`, the per-window coefficient of determination, mapping where the terrain relationship is strong and where the downscaled field rests mostly on the coarse input; `residual.sd`, the residual standard deviation of the local fit in the response's own units; and `n.valid`, the count of valid coarse cells the window held, which tells a fit that is weak because the window barely cleared the minimum valid-cell requirement apart from one that is weak because the terrain relationship is genuinely noisy there. `n.valid` is reported once per call, not once per response, since the valid-cell mask is complete-case across every response fit together. Default `FALSE`.
anomaly	Optional multi-layer coarse grid (a `SpatRaster`, `Raster*`, or `stars` object, like `data`); each layer is one period to downscale relative to `baseline`. When supplied, the result has one layer (or column) per period.
baseline	Optional single-layer coarse grid (a `SpatRaster`, `Raster*`, or `stars` object, like `data`), the coarse response baseline that `anomaly` is taken relative to. Defaults to the response layer of `data`. Ignored when `anomaly` is `NULL`.
type	`"ratio"` (multiplicative) or `"additive"`; used only with `anomaly`. Default `"ratio"`.
method	Interpolation method for bringing the coefficient grids onto `onto`. For a grid `onto`, passed to `terra::resample()`; default `"cubicspline"`. For a point (sf/SpatVector) `onto`, passed to `terra::extract()`, which only supports `"simple"` and `"bilinear"`; default `"bilinear"`. Default `NULL` uses the kind-appropriate

default; an explicit value that `terra::extract()` does not support is an error for a point target.

`output` Optional output class, one of `"terra"`, `"raster"`, `"stars"` (grid targets) or `"terra"`, `"sf"`, `"spatvector"`, `"data.frame"` (point targets). Default `NULL` returns the result in the class of `onto`.

`clamp` If `TRUE`, bound the downscaled field to the observed range of the coarse response, a guard against the local linear fit extrapolating without limit where a fine predictor lies outside the range it was fit on. Default `FALSE`.

`min_cells`, `min_variance`, `threads`
Passed to `window_regression()`. Set `threads = 1` when calling from inside a worker that is itself parallelized over targets, so the workers do not each spawn a full thread team.

Details

The relationship is given as a formula whose names refer to layers of `data` (the coarse grid) and `onto` (the target grid). Because the response and the coarse predictors are layers of a single `data` raster, they are guaranteed to share a grid; the formula names match the predictors between `data` and `onto`, so a missing or mis-ordered layer is an error rather than a silently wrong result. Only bare layer names combined with `+` are supported, such as `prec ~ elev + slope`. Transformations (`log(elev)`), interactions (`elev:slope`), and `.` are rejected; create the derived layer first.

In the common case there is one coarse response and a fine predictor such as a digital elevation model, and no coarse predictor in hand. A predictor named in the formula but absent from `data` is then derived by aggregating its `onto` layer to the response grid with `aggregate`, so `topocast(prec ~ elev, data = prec_coarse, onto = dem_fine, radius = 15)` works directly from a coarse climate layer and a fine DEM.

Several responses that share the predictors are downscaled together by naming them as `cbind(prec, tmin) ~ elev`. The moving-window design depends only on the predictors, so it is fit once and solved against every response; the result has one layer (or column) per response. With `coefficients` or `diagnostics`, the extra grids are prefixed by the response name. The setup a call does around the fit is shared the same way: the coefficient grids reach `onto` in one resampling step and the `onto` predictors are read once for the call. Responses on one coarse grid sharing one `onto`, such as several climate scenarios downscaled onto the same range, therefore cost less named together in one call than split across several.

For a time series, supply `anomaly`: a stack of coarse periods. The baseline relationship is fit once and each period's coarse anomaly, relative to `baseline`, is carried onto the fine baseline. Use `type = "ratio"` for non-negative variables such as precipitation and `type = "additive"` for variables such as temperature. With `type = "ratio"`, a coarse cell where the baseline is zero and the period is a genuine `x / 0` (undefined) is `NA`; where both the baseline and the period are zero, the ratio is taken as `1` (no change) rather than `NA`.

`data` and `onto` may be any of the common spatial classes. A gridded input is accepted as a `SpatRaster`, a `Raster*` object (raster), or a `stars` object. The target `onto` may instead be a set of points as an `sf` or `SpatVector` object, in which case the fitted relationship is evaluated at each point and a prediction column is returned; the points must carry the fine predictor values as attributes. By default the result is returned in the class of `onto`; set `output` to request another.

Value

The downscaled result on the geometry of `onto`, in the class of `onto` or the class named by `output`. For a grid target: a single layer named for the response; one layer per response with a `cbind()` left-hand side; one layer per period when `anomaly` is supplied; the fitted layer plus `(Intercept)` and slope grids when `coefficients = TRUE`; and `r.squared` and `residual.sd` grids plus a shared `n.valid` grid when `diagnostics = TRUE`. With several responses the coefficient, `r.squared`, and `residual.sd` grids are prefixed by the response name; `n.valid` is not, since it is the same valid-cell mask for every response fit in the call. For a point target the same quantities are returned as prediction columns.

See Also

`window_regression()` for the matrix engine.

Examples

```
library(terra)
set.seed(1)
coarse <- rast(nrows = 20, ncols = 20, xmin = 0, xmax = 20, ymin = 0, ymax = 20,
               crs = "EPSG:32632")
elevation <- setValues(coarse, runif(ncell(coarse), 0, 2000))
precipitation <- 800 - 0.1 * elevation
data <- c(precipitation, elevation)
names(data) <- c("prec", "elev")
terrain <- disagg(elevation, fact = 4, method = "bilinear")
names(terrain) <- "elev"

# spatial downscale
fine <- topocast(prec ~ elev, data = data, onto = terrain, radius = 4)

# one-DEM shortcut: the coarse predictor is derived from the fine DEM
fine2 <- topocast(prec ~ elev, data = data[["prec"]], onto = terrain, radius = 4)

# return the local coefficient grids
coef_grids <- topocast(prec ~ elev, data = data, onto = terrain, radius = 4,
                       coefficients = TRUE)

# several responses sharing the predictor, downscaled in one call
temperature <- 25 - 0.006 * elevation
data2 <- c(precipitation, temperature, elevation)
names(data2) <- c("prec", "temp", "elev")
both <- topocast(cbind(prec, temp) ~ elev, data = data2, onto = terrain, radius = 4)

# diagnostics: the local fit quality as an r.squared grid
with_r2 <- topocast(prec ~ elev, data = data, onto = terrain, radius = 4,
                    diagnostics = TRUE)

# time series: supply the periods
months <- precipitation * c(0.8, 1.2)
names(months) <- c("jan", "feb")
series <- topocast(prec ~ elev, data = data, onto = terrain, radius = 4,
```

```
                  anomaly = months, type = "ratio")

# predict at point locations: onto is sf points carrying the predictor
if (requireNamespace("sf", quietly = TRUE)) {
  plots <- sf::st_as_sf(
    data.frame(x = c(5, 10, 15), y = c(5, 10, 15), elev = c(500, 1000, 1500)),
    coords = c("x", "y"), crs = "EPSG:32632")
  at_plots <- topocast(prec ~ elev, data = data, onto = plots, radius = 4)
}
```

`window_regression` *Moving-window linear regression over raster matrices*

Description

Fits, for every cell, a linear regression of one or more responses on a shared set of predictors over a square moving window, and returns full-resolution intercept and slope grids together with a per-cell coefficient of determination. The fit uses summed-area tables, so the cost does not grow with the window size. Cells are excluded from a window where any response or predictor is non-finite; a cell is returned as `NA` when its window holds fewer valid cells than the model needs or a predictor has no spread. If every cell of every response and predictor is non-finite, there is nothing to regress and the call errors instead of returning an all-`NA` result.

Usage

```
window_regression(
  y,
  x,
  radius,
  min_cells = 0L,
  min_variance = 1e-08,
  threads = NULL
)
```

Arguments

`y`	Numeric matrix, the response on the coarse grid, or a list of numeric matrices for several responses sharing the predictors and grid.
`x`	Numeric matrix, or a list of numeric matrices, the predictor(s) on the same grid as `y`.
`radius`	Integer window radius in cells, at least `1`; the window is a square of side `2 * radius + 1`. A radius of `0` is rejected: its one-cell window can never hold the `k + 1` valid cells a fit needs.
`min_cells`	Integer, additional valid cells required in a window beyond the `k + 1` model terms (`k` predictors plus the intercept). Default `0`.

`min_variance` Numeric, the minimum within-window variance a predictor must have for the cell to be fit. Default `1e-8`.

`threads` Integer, the number of threads the fitting loop may use, or `NULL` (the default) for every core available. A request above the number of cores the OpenMP runtime offers is clamped to that number, and every request is clamped to two while a package check is running. Set `threads = 1` when calling from inside a worker that is itself parallelized over targets, so the workers do not each spawn a full thread team; the fit is a small share of a call, so more workers at one thread each generally use the cores better than one worker at many. Without OpenMP the fit is single-threaded and this argument has no effect.

Details

Several responses share the predictors, so the window design is assembled and factored once and solved against every response, and each extra response costs only a back-substitution. The valid-cell mask is therefore complete-case across the responses: a cell enters a window only where every response and predictor is finite.

This is the matrix engine behind `topocast()`; it works on plain numeric matrices and does not depend on terra.

Value

A list with `intercept` (a numeric matrix), `slope` (a list of numeric matrices, one per predictor), `r_squared` (a numeric matrix), `residual_sd` (a numeric matrix, the residual standard deviation of the local fit; `NA` where the window has no residual degrees of freedom), and `n_valid` (a numeric matrix, the count of valid cells the window held), each the same size as `y`. `n_valid` is shared across responses, since the valid-cell mask is complete-case across them. When `y` is a list of responses, `intercept`, `r_squared`, and `residual_sd` are lists of matrices and `slope` is a list of per-response slope lists, named for the responses.

See Also

`topocast()` for the terra workflow.

Examples

```
set.seed(1)
elevation <- matrix(runif(100, 0, 1000), 10, 10)
climate <- 50 + 0.01 * elevation
fit <- window_regression(climate, elevation, radius = 3)
str(fit)

# several responses sharing the predictor: the design is factored once
rain <- 800 - 0.1 * elevation
temp <- 15 - 0.006 * elevation
both <- window_regression(list(rain = rain, temp = temp), elevation, radius = 3)
names(both$slope)
```

Index

terra::extract(), *3*
terra::resample(), *3*
topocast, 2
topocast(), *7*

window_regression, 6
window_regression(), *4*, *5*