Package ‘stgam’

January 29, 2026

Title Spatially and Temporally Varying Coefficient Models Using Generalized Additive Models

Version 1.2.0

Author Lex Comber [aut, cre],
Paul Harris [ctb],
Gonzalo Irisarri [ctb],
Chris Brunsdon [ctb]

Maintainer Lex Comber <`a.comber@leeds.ac.uk`>

Description A framework for undertaking space and time varying coefficient models (varying parameter models) using a Generalized Additive Model (GAM) with smooths approach. The framework suggests the need to investigate for the presence and nature of any space-time dependencies in the data. It proposes a workflow that creates and refines an initial space-time GAM and includes tools to create and evaluate multiple model forms. The workflow sequence is to: i) Prepare the data by lengthening it to have a single location and time variables for each observation. ii) Create all possible space and/or time models in which each predictor is specified in different ways in smooths. iii) Evaluate each model via their AIC value and pick the best one. iv) Create the final model. v) Calculate the varying coefficient estimates to quantify how the relationships between the target and predictor variables vary over space, time or space-time. vi) Create maps, time series plots etc. The number of knots used in each smooth can be specified directly or iteratively increased. This is illustrated with a climate point dataset of the dry rain forest in South America. This builds on work in Comber et al (2024) <doi:10.1080/13658816.2023.2270285> and Comber et al (2004) <doi:10.33

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Suggests cols4all, knitr, ggplot2, cowplot, rmarkdown, testthat (>= 3.0.0), tidyr, lubridate, sf, gratia, kableExtra

Config/testthat/edition 3

URL `https://github.com/lexcomber/stgam`

BugReports `https://github.com/lexcomber/stgam/issues`

Depends R (>= 4.1.0), mgcv (>= 1.9-1), glue

LazyData true

Imports foreach, doParallel, parallel, dplyr, magrittr, purrr, stringr

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2026-01-29 12:50:02 UTC

Contents

`calculate_vcs` *Extracts varying coefficient estimates (for SVC, TVC and STVC models).*

Description

Extracts varying coefficient estimates (for SVC, TVC and STVC models).

Usage

```
calculate_vcs(input_data, mgcv_model, terms = NULL)
```

Arguments

`input_data`	the data used to create the GAM model in `data.frame`, `tibble` or `sf` format. This can be the original data used to create the model or another surface with location and time attributes.
`mgcv_model`	a GAM model with smooths created using the `mgcv` package
`terms`	a vector of names starting with "Intercept" plus the names of the covariates used in the GAM model (these are the names of the variables in the `input_data` used to construct the model).

Value

A `data.frame` of the input data, the coefficient estimates, the standard errors and the t-values estimates for each covariate. It can be used to generate coefficient estimates for specific time slices and over gridded surfaces as described in the package vignette.

Examples

```
require(dplyr)
require(doParallel)
# define input data
data("chaco")
input_data <-
  chaco |>
  # create Intercept as an addressable term
  mutate(Intercept = 1)
# create a model for example as result of running `evaluate_models`
gam.m = gam(ndvi ~ 0 + s(X, Y, by = Intercept) +
 s(X, Y, by = tmax) + s(X, Y, by = pr), data = input_data)
# calculate the Varying Coefficients
terms = c("Intercept", "tmax", "pr")
vcs = calculate_vcs(input_data, gam.m, terms)
vcs |> select(ndvi, X, Y, starts_with(c("b_", "se_", "t_")), yhat)
```

`chaco` *Chaco dry rainforest data (2012-2022)*

Description

A point dataset of NDVI and climate data. The data are sample of 2000 observations of Normalised Difference Vegetation Index (NDVI) (2012-2022) of the Chaco dry rainforest in South America with some climate data. These are found via Google Earth Engine (Gorelick et al., 2017). The NDVI data is sourced from the PKU GIMMS NDVI v1.2 dataset, which provides NDVI observations at 1/12° spatial resolution at bi-monthly intervals from 1982 to 2022 (Li et al., 2023). The climate data was derived from the TerraClimate dataset (IDAHO_EPSCOR/TERRACLIMATE). Maximum temperature (`tmax`) and Precipitation (`pr`) were selected and means calculated for each monthly image across all pixels.

Usage

```
chaco
```

Format

A `sf` POINT dataset with 2000 observations and 12 fields.

id An observation identifier

ndvi Normalised Difference Vegetation Index (NDVI)

tmax Maximum temperature (°C)

pr Preciptation

month A continous integer variable from 1 to 120

year The year of observation

lon Longitude in degrees (WGS84)

lat Latitude in degrees (WGS84)

X Easting in metres from the SIRGAS 2000 / Brazil Mercator projection (EPSG:5641)

Y Northing in metres from the SIRGAS 2000 / Brazil Mercator projection (EPSG:5641)

geometry The spatial geometry of the observation in the SIRGAS 2000 / Brazil Mercator projection (EPSG:5641)

Source

Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D., & Moore, R. (2017). Google Earth Engine: Planetary-scale geospatial analysis for everyone. Remote Sensing of Environment, 202, 18–27. https://doi.org/10.1016/J.RSE.2017.06.031

Li, M., Cao, S., Zhu, Z., Wang, Z., Myneni, R. B., & Piao, S. (2023). Spatiotemporally consistent global dataset of the GIMMS Normalized Difference Vegetation Index (PKU GIMMS NDVI) from 1982 to 2022. Earth System Science Data, 15(9), 4181–4203. https://doi.org/10.5194/ESSD-15-4181-2023

Examples

```
library(sf)
data("chaco")
```

`effect_size` *Quantifies the relative effect sizes of each component of zmgcv‘ GAM model.*

Description

Quantifies the relative effect sizes of each component of zmgcv‘ GAM model.

Usage

```
effect_size(mgcv_model, digits = 3)
```

Arguments

`mgcv_model`	a GAM model created by the `mgcv` package
`digits`	the number of significant figures of uysed to report effect size

Value

a matrix of the model terms, the size of the effect (range) ad standard deviation (sd)

Examples

```
require(dplyr)
require(stringr)
require(purrr)
require(doParallel)

# define input data
data("chaco")
m <- gam(
ndvi ~
  te(X,Y, by = tmax) +
  s(X,Y, by = pr),
 data = chaco,
 method = "REML",
 family = gaussian()
)
# examine the effect size
effect_size(m, 3)
```

---

evaluate_models — *Evaluates multiple models with each predictor variable specified in different ways in order to determining model form*

Description

Evaluates multiple models with each predictor variable specified in different ways in order to determining model form

Usage

```
evaluate_models(
  input_data,
  target_var,
  model_family = "gaussian()",
  vars,
  coords_x = "X",
  coords_y = "Y",
  VC_type = "SVC",
  time_var = NULL,
  k_set = FALSE,
  spatial_k = 50,
  temporal_k = 10,
  k_increase = FALSE,
  k2edf_ratio = 1.5,
  k_multiplier = 2,
  max_iter = 10,
  ncores = 2
)
```

Arguments

`input_data`	he data to be used used to create the GAM model in (`data.frame` or `tibble` format), containing an Intercept column to allow it be treated as an addressable term in the model.
`target_var`	the name of the target variable.
`model_family`	the mdoel family, defaults to Guassian
`vars`	a vector of the predictor variable names (without the Intercept).
`coords_x`	the name of the X, Easting or Longitude variable in `input_data`.
`coords_y`	the name of the Y, Northing or Latitude variable in `input_data`.
`VC_type`	the type of varying coefficient model: options are "TVC" for temporally varying, "SVC" for spatially varying and "STVC" for space-time.
`time_var`	the name of the time variable if undertaking STVC model evaluations.
`k_set`	a logical value for user defined `k` values. The default is `FALSE`. Cannot be used with `k_increase`.
`spatial_k`	the value of `k` for spatial smooths if `k_set` is `TRUE`.
`temporal_k`	the value of `k` for temporal smooths if `k_set` is `TRUE`.
`k_increase`	a logical value of whether to check and increase the number of knots in each smooth. The default is `FALSE`.
`k2edf_ratio`	a threshold of the ratio of the number of knots, `k`, in each smooth to its Effective Degrees of Freedom. If any smooth has a *knots-to-EDF* ratio less than this value then the knots are iteratively increased by the `k_multiplier` value until the threshold check is passed, the number knots passes the maximum degrees of freedom, or the number of iterations, `max_iter` is reached. Cannot be used with `k_set`.
`k_multiplier`	a multiplier by which the knots are increased on each iteration. The default is 2.
`max_iter`	the maximum number of iterations that `k` is increased.
`ncores`	the number of cores to use in parallelised approaches (default is 2 to overcome CRAN package checks). This can be determined for your computer by running parallel::detectCores()-1. Parallel approaches are only undertaken if the number of models to evaluate is greater than 30.

Value

a `data.frame` with indices for each predictor variable, the knots specified in each smooth (`ks`), a AIC score (`aic`) for each model and the associated formula (`f`). The output should be passed to the `gam_model_rank` function.

Examples

```
## Not run:
require(dplyr)
require(doParallel)
require(sf)
```

```
# define input data
data("chaco")
input_data <-
  chaco |>
  # create Intercept as an addressable term
  mutate(Intercept = 1) |>
  # remove the geometry
  st_drop_geometry()

# evaluate different model forms
# example 1 with 6 models and no `k` adjustment
svc_mods <-
  evaluate_models(
    input_data = input_data,
    target_var = "ndvi",
    model_family = "gaussian()",
    vars = c("tmax"),
    coords_x = "X",
    coords_y = "Y",
    VC_type = "SVC"
  )
# have a look!
svc_mods

# example 2 with 6 models and `k` adjustment
svc_k1_mods <-
  evaluate_models(
    input_data = input_data,
    target_var = "ndvi",
    vars = c("tmax"),
    model_family = "gaussian()",
    coords_x = "X",
    coords_y = "Y",
    VC_type = "SVC",
    k_increase = TRUE,
    k2edf_ratio = 1.5,
    k_multiplier = 2,
    max_iter = 10
  )
# have a look!
svc_k1_mods

# example 3 with 6 models and `k` set by user
svc_k2_mods <-
  evaluate_models(
    input_data = input_data,
    model_family = "gaussian()",
    target_var = "ndvi",
    vars = c("tmax"),
    coords_x = "X",
    coords_y = "Y",
    VC_type = "SVC",
    time_var = NULL,
    k_set = TRUE,
    spatial_k = 20,
  )
# have a look!
svc_k2_mods

## End(Not run)
```

`gam_model_rank` *Ranks models by AIC, giving the model form for each predictor variable.*

Description

Ranks models by AIC, giving the model form for each predictor variable.

Usage

```
gam_model_rank(res_tab, n = 10)
```

Arguments

`res_tab`	a `data.frame` returned from the `evaluate_models()` function.
`n`	the number of ranked models to return.

Value

a `tibble` of the 'n' best models, ranked by AIC, with the form of each predictor variable where '—' indicates the absence of a predictor, 'Fixed' that a parametric form was specified, 's(S)' a spatial smooth, 's(T)' a temporal smooth and 'te(ST)' a combined space-time smooth. Model AIC is reported as are the knots in each smooth (`ks`) and the formula of each model (`f`).

Examples

```
require(dplyr)
require(stringr)
require(purrr)
require(doParallel)
require(sf)

# define input data
data("chaco")
input_data <-
  chaco |>
  # create Intercept as an addressable term
  mutate(Intercept = 1) |>
  # remove the geometry
  st_drop_geometry()
```

```
# evaluate different model forms
# example 1 with 6 models and no `k` adjustment
svc_mods <-
  evaluate_models(
    input_data = input_data,
    target_var = "ndvi",
    vars = c("tmax"),
    coords_x = "X",
    coords_y = "Y",
    VC_type = "SVC"
  )
# rank the models
gam_model_rank(svc_mods)
```

Index