Package 'stgam'

July 31, 2024

Title Spatially and Temporally Varying Coefficient Models Using Generalized Additive Models

Version 0.0.1.2

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

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

Description A framework for specifying spatially, temporally and spatially-and-temporally varying coefficient models using Generalized Additive Models with Gaussian Process smooths. The smooths are parameterised with location and / or time attributes. Importantly the framework supports the investigation of the presence and nature of any space-time dependencies in the data, allows the user to evaluate different model forms (specifications) and to pick the most probable model or to combine multiple varying coefficient models using Bayesian Model Averaging. For more details see: Brunsdon et al (2023) <doi:10.4230/LIPIcs.GIScience.2023.17>, Comber et al (2023) <doi:10.4230/LIPIcs.GIScience.2023.22> a

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.3

Suggests cols4all, knitr, purrr, rmarkdown, sf, testthat (>= 3.0.0), tidyr

Config/testthat/edition 3

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

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

Depends R (>= 2.10),

LazyData true

Imports cowplot, doParallel, dplyr, foreach, ggplot2, glue, grDevices, magrittr, metR, mgcv (>= 1.9-1), parallel, tidyselect

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2024-07-31 14:30:02 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, model, terms)
```

Arguments

`input_data`	the data used to create the GAM model in `data.frame`, `tibble` or `sf` format
`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 `data`)

Value

A `data.frame` of the input data and the coefficient and standard error estimates for each covariate.

Examples

```
library(dplyr)
library(mgcv)
# SVC
data(productivity)
input_data = productivity |> dplyr::filter(year == "1970") |> mutate(Intercept = 1)
gam.svc.mod = gam(privC ~ 0 + Intercept +
                  s(X, Y, bs = 'gp', by = Intercept) +
                  unemp + s(X, Y, bs = "gp", by = unemp) +
                  pubC + s(X, Y, bs = "gp", by = pubC),
                  data = input_data)
terms = c("Intercept", "unemp", "pubC")
```

```
svcs = calculate_vcs(input_data, gam.svc.mod, terms)
head(svcs)
```

---

do_bma *Undertake undertake coefficient averaging using Bayesian Model Avergaing (BMA), weighting different models by their probabilities*

Description

Undertake undertake coefficient averaging using Bayesian Model Avergaing (BMA), weighting different models by their probabilities

Usage

```
do_bma(model_table, terms, thresh = 0.1, relative = FALSE, input_data)
```

Arguments

`model_table`	a table of competing models generated by the `gam_model_probs()` function
`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 `data`)
`thresh`	a probability threshold value above which to combine competing models
`relative`	logical to indicate whether the probabilities in `data` are relative (`Pr(M)`) or absolute (`Pr(M|D)`)
`input_data`	the input data with a named Intercept term, in `data.frame`, `tibble` or `sf` format

Value

A `matrix` of the probability weighted averaged coefficient estimates from multiple models.

Examples

```
library(cols4all)
library(dplyr)
library(sf)
library(glue)
library(purrr)
library(mgcv)
library(sf)
library(ggplot2)
# data
data(productivity)
input_data = productivity |> filter(year == "1970") |> mutate(Intercept = 1)
# create and evaluate multiple models
svc_res_gam = evaluate_models(input_data, STVC = FALSE)
# determine their probabilities
mod_comp_svc <- gam_model_probs(svc_res_gam)
# combine the model coefficients
```

```
svc_bma <- do_bma(mod_comp_svc,
                  terms = c("Intercept", "unemp", "pubC"),
                  thresh = 0.1,
                  relative = FALSE,
                  input_data = input_data)
head(svc_bma)
# join back to spatial layer
data(us_data)
svc_bma_sf <-
us_data |>
select(GEOID) |>
left_join(productivity |>
  filter(year == "1970") |>
  select(GEOID, year) |>
  cbind(svc_bma)) |>
  relocate(geometry, .after = last_col())
# and map
tit =expression(paste(""*beta[`Public Capital`]*" "))
ggplot(data = svc_bma_sf, aes(fill=pubC)) +
  geom_sf() +
  scale_fill_continuous_c4a_div(palette="brewer.yl_or_rd",name=tit) +
  coord_sf() +
  theme_linedraw()
```

`evaluate_models` *Creates and evaluates multiple varying coefficient GAM GP smooth models (SVC or STVC)*

Description

Creates and evaluates multiple varying coefficient GAM GP smooth models (SVC or STVC)

Usage

```
evaluate_models(
  input_data,
  target_var = "privC",
  covariates = c("unemp", "pubC"),
  coords_x = "X",
  coords_y = "Y",
  STVC = FALSE,
  time_var = NULL,
  ncores = 2
)
```

Arguments

`input_data`	a `data.frame`, `tibble` `sf` containing the target variables, covariates and coordinate variables

target_var	the name of the target variable in data
covariates	the name of the covariates (predictor variables) in data
coords_x	the name of the X, Easting or Longitude variable in data
coords_y	the name of the Y, Northing or Latitude variable in data
STVC	a logical operator to indicate whether the models Space-Time (TRUE) or just Space (FALSE)
time_var	the name of the time variable if undertaking STVC model evaluations
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 table in data.frame format of all possible model combinations with each covariate specified in all possible ways, with the BIC of the model and the model formula.

Examples

```
library(dplyr)
library(glue)
library(purrr)
library(doParallel)
library(mgcv)
data("productivity")
input_data = productivity |> filter(year == "1970")
svc_res_gam =
  evaluate_models(input_data = input_data,
                  target_var = "privC",
                  covariates = c("unemp", "pubC"),
                  coords_x = "X",
                  coords_y = "Y",
                  STVC = FALSE)
head(svc_res_gam)
```

---

gam_model_probs *Calculates the model probabilities of the different GAM models generated by evaluate_models‘*

Description

Calculates the model probabilities of the different GAM models generated by evaluate_models‘

Usage

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

Arguments

res_tab	a table generated by evaluate_models
n	the number of models to retain and generate probabilities for

Value

A ranked data table in tibble format of the top n models, their form, BIC and model or (Pr(M|D)) or relative (Pr(M)) probability value. Model probability indicates the probability of the each model being the correct model and the relative probabilities provide a measure of the doubt about the differences in model specification, when compared to the best or highest ranked model. The relative probabilities are needed when large BIC values generate near zero probability values.

Examples

```
library(dplyr)
library(purrr)
library(glue)
library(mgcv)
data(productivity)
input_data = productivity |> filter(year == "1970")
svc_res_gam = evaluate_models(input_data, STVC = FALSE)
mod_comp_svc <- gam_model_probs(svc_res_gam, n = 10)
# print out the terms and probabilities
mod_comp_svc|> select(-f)
```

---

plot_1d_smooth *Plots a 1-Dimensional GAM smooth*

Description

Plots a 1-Dimensional GAM smooth

Usage

```
plot_1d_smooth(mod, ncol = NULL, nrow = NULL, fills = "lightblue")
```

Arguments

mod	a GAM model with smooths created using the mgcv package
ncol	the number of columns in the compound plot
nrow	the number of rows in the compound plot
fills	the fill colours (single or vector)

Value

A compound plot of the GAM 1-dimensional smooths (rendered using cowplot::plot_grid()).

Examples

```
library(mgcv)
library(ggplot2)
library(dplyr)
library(cowplot)
# 1. from the `mgcv` `gam` function help
set.seed(2) ## simulate some data...
dat <- gamSim(1,n=400,dist="normal",scale=2)
b <- gam(y~s(x0)+s(x1)+s(x2)+s(x3),data=dat)
plot_1d_smooth(b, ncol = 2, fills = c("lightblue", "lightblue3"))
dev.off()
# 2. using a TVC
data(productivity)
data = productivity |> mutate(Intercept = 1)
gam.tvc.mod = gam(privC ~ 0 + Intercept +
                  s(year, bs = 'gp', by = Intercept) +
                  unemp + s(year, bs = "gp", by = unemp) +
                  pubC + s(year, bs = "gp", by = pubC),
                  data = data)
plot_1d_smooth(gam.tvc.mod, fills = "lightblue")
```

`plot_2d_smooth` *Plots a 2-Dimensional GAM smooth*

Description

Plots a 2-Dimensional GAM smooth

Usage

```
plot_2d_smooth(mod, filled = FALSE, outline = NULL, ncol = NULL, nrow = NULL)
```

Arguments

`mod`	a GAM model with smooths created using the `mgcv` package
`filled`	`logical` value to indicate whether a filled plot should be created (`TRUE`) or not (`FALSE`)
`outline`	the name of an `sf` object to be plotted (NULL is the default)
`ncol`	the number of columns for the compound plot
`nrow`	the number of rows for the compound plot

Value

A compound plot of the 2-dimensional smooths (rendered using `cowplot::plot_grid`).

Examples

```
library(mgcv)
library(ggplot2)
library(dplyr)
library(metR)
library(cowplot)
set.seed(2) ## simulate some data...
dat <- gamSim(1,n=400,dist="normal",scale=2)
# use x1 and x2 as the coordinates
b <- gam(y~s(x0, x1, bs = 'gp', by = x2),data=dat)
plot_2d_smooth(b, filled = TRUE)
```

`productivity` *US States Economic Productivity Data (1970-1985)*

Description

A dataset of annual economic productivity data for the 48 contiguous US states (with Washington DC merged into Maryland), from 1970 to 1985 (17 years) in long format. The data productivity data table was extracted from the `plm` package.

Usage

```
productivity
```

Format

A tibble with 816 rows and 14 columns.

state The name of the state

GEOID The state code

region The region

pubC Public capital which is composed of highways and streets (hwy) water and sewer facilities (water) and other public buildings and structures (util)

hwy Highway and streets assets

water Water utility assets

util Other public buildings and structures

privC Private captial stock

gsp Gross state product

emp Labour input measured by the employment in non-agricultural payrolls

unemp State unemployment rate capture elements of the business cycle

X Easting in metres from USA Contiguous Equidistant Conic projection (ESRI:102005)

Y Northing in metres from USA Contiguous Equidistant Conic projection (ESRI:102005)

Source

Croissant, Yves, Giovanni Millo, and Kevin Tappe. 2022. Plm: Linear Models for Panel Data

Examples

```
data(productivity)
```

---

us_data *US States boundaries*

Description

A dataset of of the boundaries of 48 contiguous US states (with Washington DC merged into Maryland), extracted from the `spData` package.

Usage

```
us_data
```

Format

A `sf` polygon dataset with 48 rows and 6 fields.

GEOID The state code

NAME The name of the state

REGION The region

total_pop_10 Population in 2010

total_pop_15 Population in 2015

Source

Bivand, Roger, Jakub Nowosad, and Robin Lovelace. 2019. spData: Datasets for Spatial Analysis. R package

Examples

```
data(us_data)
```

Index