Package ‘stfit’

October 2, 2020

Type Package

Title Spatio-Temporal Functional Imputation Tool

Version 0.99.8

Date 2020-09-29

Description A general spatiotemporal satellite image imputation method based on sparse functional data analytic techniques. The imputation method applies and extends the Functional Principal Analysis by Conditional Estimation (PACE). The underlying idea for the proposed procedure is to impute a missing pixel by borrowing information from temporally and spatially contiguous pixels based on the best linear unbiased prediction.

BugReports https://github.com/mingsnu/stfit/issues

License GPL-3

LazyData true

Depends R (>= 3.5.0)

Imports Rcpp, Matrix, doParallel, foreach, abind, fda, raster, rasterVis, RColorBrewer

LinkingTo Rcpp

Suggests testthat, dplyr

RoxygenNote 7.1.0

Encoding UTF-8

NeedsCompilation yes

Author Weicheng Zhu [aut, cre]

Maintainer Weicheng Zhu <mingsnu@gmail.com>

Repository CRAN

Date/Publication 2020-10-02 08:30:02 UTC

R topics documented:

stfit-package ................................................................. 2
ARE ............................................................................. 2
stfit-package

stfit: Spatial-Temporal Functional Imputation Tool

Description

The stfit package provides functions to impute missing values for a sequence of observed images for the same location using functional data analysis technique.

ARE

Absolute relative error

Description

Absolute relative error

Usage

ARE(y, ypred)
Arguments

- y vector
- ypred vector

Value

numeric number. A measure of difference between y and ypred.

---

**epan**

*Epanicov kernel function*

Description

Epanicov kernel function

Usage

epan(x)

Arguments

- x numeric vector

Value

vector

---

**getMask**

*Get image mask*

Description

Get image mask

Usage

getMask(object, tol = 0.95)

Arguments

- object A numeric matrix. Each row is an row stacked image.
- tol If the percentage of missing values for a pixel over time is greater than this value, this pixel is treated as a mask value.
**getMissingLayers**  
*Get missing layer index*

**Description**
Get missing layer index

**Usage**
```r
getMissingLayers(rst.list)
```

**Arguments**
- `rst.list` a RasterStack or RasterBrick object or a list of them

**Value**
index of the missing layers

---

**landsat106**  
*Landsat data example*

**Description**
A dataset containing observation values of a 31x31 pixcels landsat image observed between year 1982 and 2015.

**Usage**
```r
landsat106
```

**Format**
A data frame with 990 rows and 963 columns:
- year
- doy day of the year
- pixeli pixel value for the i-th pixel of the image

An object of class `tbl_df` (inherits from `tbl_data.frame`) with 990 rows and 963 columns.
landsatVis  

Data visualization for landsat data

Description

Data visualization for landsat data

Usage

landsatVis(
    mat,
    img.nrow = 31,
    byrow = FALSE,
    colthm = rasterTheme(panel.background = list(col = "black"),
                         region = brewer.pal(9, "YlOrRd"),
                         ...
)
)

Arguments

- **mat**: A matrix, each row corresponds to a vectorized image pixel values.
- **img.nrow**: number of rows of the image
- **byrow**: logical value indicating whether the pixel values are stored by row or by column. Default to FALSE
- **colthm**: Color theme for the plot, passing to the par.settings parameter of the levelplot function in the rasterVis package
- ...: All other options passed to levelplot function in the rasterVis package

Examples

landsatVis(landsat106[landsat106$year == 2015, -c(1:2)],
           names.attr = as.character(landsat106$doy[landsat106$year == 2015]))

lc_cov_1d  

Local constant covariance estimation

Description

Local constant covariance estimation

Usage

lc_cov_1d(ids, time, resid, W, t1, t2)
Arguments

ids a vector indicating subject/group ids
time integer vector of observed time points, the minimum time unit is 1
resid vector of residual values used for covariance calculation
W weight vector, it contains both kernel and bandwidth information in general local polynomial estimation setting up
t1 time point 1
t2 time point 2

lc_cov_1d_est Local constant covariance estimation

Description

Local constant covariance estimation

Usage

lc_cov_1d_est(ids, time, resid, W, tt)

Arguments

ids a vector indicating subject/group ids
time integer vector of observed time points, the minimum time unit is 1
resid vector of residual values used for covariance calculation
W weight vector, it contains both kernel and bandwidth information in general local polynomial estimation setting up
tt time vector

llreg Local linear regression

Description

Local linear regression

Usage

llreg(x, y, x.eval = x, minimum.num.obs = 4, h = 60, Kern = epan)
**Arguments**

- `x` independent variable
- `y` response variable
- `x.eval` new data to predict on
- `minimum.num.obs` minimum number of observations needed to run the regression
- `h` bandwidth
- `Kern` Kernel

**Value**

predicted values at `x.eval`

---

**Description**

Local Polynomial Regression

**Usage**

```r
lpreg(x, y, x.eval, minimum.num.obs = 4, span = 0.3, ...)
```

**Arguments**

- `x` independent variable
- `y` response variable
- `x.eval` vector to predict on
- `minimum.num.obs` minimum number of observations needed to run the regression
- `span` see `loess` function
- `...` other parameters passed to `loess` function

**Value**

predicted values at `x.eval`
meanEst  

STFIT Mean Estimation

Description

The function is used for pixel-wise mean estimation.

Usage

```r
meanEst(
  doy,
  mat,
  doyeval = seq(min(doy), max(doy)),
  msk = rep(FALSE, ncol(mat)),
  outlier.tol = 0.5,
  minimum.num.obs = 4,
  cluster = NULL,
  redo = TRUE,
  clipRange = c(-Inf, Inf),
  clipMethod = c("truncate", "nnr"),
  img.nrow = NULL,
  img.ncol = NULL
)
```

Arguments

doy  
- vector of day of year (DOY) index

mat  
- data matrix. Each row contains a row stacked image pixel values.

doyeval  
- a vector of DOY on which to get the mean imputation

msk  
- an optional logistic vector. TRUE represent the corresponding pixel is always missing.

outlier.tol  
- the tolerance value in defining an image as outlier. The percent of outlier pixels in an image exceed this value is regarded as outlier image which will not be used in temporal mean estimation.

minimum.num.obs  
- minimum number of observations needed for mean estimation. Too few observations may lead to big estimation error.

cluster  
- an optional vector defining clusters of pixels. If NULL, mean estimation is conducted on each pixel, otherwise all pixels from the same cluster are combined for mean estimation.

redo  
- whether to recalculate the mean estimation if there is an outlier (only redo once).

clipRange  
- vector of length 2, specifying the minimum and maximum values of the prediction value

clipMethod  
- "nnr" or "truncate". "nnr" uses average of nearest neighbor pixels to impute; "truncate use the clipRange value to truncate."
**Details**

There are several predefined methods for mean estimation: `smooth_spline`, `llreg`, `lpreg` and `spreg`. User can use `opt$get()` to check the current registered method and use `opt$set()` function to set the method. For example, one can run `opt$set(smooth_spline)` first and then run the `meanEst` function to use smoothing spline regression for mean estimation. User can also customize the methods for mean estimation. For example, mean estimation through fourier basis expansion:

```r
.X = fda::eval.basis(1:365, fda::create.fourier.basis(rangeval=c(0,365), nbasis=11))
customfun <- function(x, y, x.eval=1:365, minimum.num.obs = 10){
  nonna.idx = !is.na(y)
  if(sum(nonna.idx) < minimum.num.obs)
    return(rep(NA, 365))
  ## lmfit = lm.fit(.X[unlist(lapply(x, function(x) which(x == x.eval))),], y[nonna.idx])
  lmfit = lm.fit(.X[x[nonna.idx],], y[nonna.idx])
  return(.X[x.eval,]
}
stfit::opts_stfit$set(temporal_mean_est = customfun)
```

**Value**

A list containing the following entries:

- `doyeval`: same as input `doyeval`
- `meanmat`: estimated mean matrix, with number of rows equals length of `doyeval` and number of columns equal `ncol(mat)`
- `idx`: a list of image indexes
  - `idx.allmissing`: completely missing image indexes,
  - `idx.partialmissing`: partially observed image indexes,
  - `idx.fullyobserved`: fully observed image indexes,
  - `idx.outlier`: outlier image indexes.
- `outlier`: a list of image outliers information
  - `outidx`: index of the outlier image
  - `outpct`: percentage of outlier pixels corresponding to `outidx`
  - `outlst`: a list of the same length as `outidx`, with each list the missing pixel index.
**Description**

Normalized Mean Square Estimation

**Usage**

\[
\text{NMSE}(y, \text{ypred})
\]

**Arguments**

- \( y \) vector
- \( \text{ypred} \) vector

**Value**

numeric number. A measure of difference between \( y \) and \( \text{ypred} \).

---

**opts_stfit** Options for stfit

**Description**

Options for stfit

**Usage**

\[
\text{opts_stfit}
\]

**Format**

An object of class \text{list} of length 3.
outlier

*Image Outlier Detection*

**Description**

Image Outlier Detection

**Usage**

`outlier(mat)`

**Arguments**

- `mat` data matrix. Each row is a row stacked image.

**Value**

a list containing the following entries:

- `outidx`: index of the outlier image
- `outpct`: percentage of outlier pixels corresponding to `outidx`
- `outlst`: a list of the same length as `outidx`, with each element the missing pixel index.

**Examples**

```r
dfB = landsat106[landsat106$year >= 2000,]
matB = as.matrix(dfB[,c(1:2)])
outlier(matB)
```

pctMissing

*Missing value percentages*

**Description**

Missing value percentages

**Usage**

`pctMissing(x, mc.cores)`

**Arguments**

- `x` A RasterStack object
- `mc.cores` Number of cores to use

**Value**

A vector of percent of missing values for each layer
rmOutlier

Remove outlier

Description
An outlier is defined as points outside the whiskers of the boxplot over the time domain (DOY).

Usage
rmOutlier(rst)

Arguments
rst: a *Raster object

Value
a *Raster object

RMSE

Root Mean Square Estimation

Description
Root Mean Square Estimation

Usage
RMSE(y, ypred)

Arguments
y: vector
ypred: vector

Value
numeric number. A measure of difference between y and ypred.
seffEst  

STFIT Spatial Effect Estimation

Description

STFIT Spatial Effect Estimation

Usage

seffEst(
  rmat,
  img.nrow,
  img.ncol,
  h.cov = 2,
  h.sigma2 = 2,
  weight.cov = NULL,
  weight.sigma2 = NULL,
  nnr,
  method = c("lc", "emp"),
  partial.only = TRUE,
  pve = 0.99,
  msk = NULL,
  msk.tol = 0.95,
  var.est = FALSE
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rmat</td>
<td>residual matrix</td>
</tr>
<tr>
<td>img.nrow</td>
<td>image row dimension</td>
</tr>
<tr>
<td>img.ncol</td>
<td>image column dimension</td>
</tr>
<tr>
<td>h.cov</td>
<td>bandwidth for spatial covariance estimation; ignored if weight.cov is supplied</td>
</tr>
<tr>
<td>h.sigma2</td>
<td>bandwidth for sigma2 estimation</td>
</tr>
<tr>
<td>weight.cov</td>
<td>weight matrix for spatial covariance estimation</td>
</tr>
<tr>
<td>weight.sigma2</td>
<td>weight vector for spatial variance estimation</td>
</tr>
<tr>
<td>nnr</td>
<td>maximum number of nearest neighbor pixels to use for spatial covariance estimation</td>
</tr>
<tr>
<td>method</td>
<td>&quot;lc&quot; for local constant covariance estimation and &quot;emp&quot; for empirical covariance estimation</td>
</tr>
<tr>
<td>partial.only</td>
<td>calculate the spatial effect for partially observed images only, default is TRUE</td>
</tr>
<tr>
<td>pve</td>
<td>percent of variance explained of the selected eigen values. Default is 0.99.</td>
</tr>
<tr>
<td>msk</td>
<td>an optional logistic vector. TRUE represent the corresponding pixel is always missing.</td>
</tr>
</tbody>
</table>
msk.tol  if 'msk' is not given, the program will determine the mask using getMask function. If the percentage of missing values for a pixel over time is greater than this

var.est  Whether to estimate the variance of the temporal effect. Default is FALSE.

Value

List of length 3 with entries:

- seff_mat: estimated spatial effect matrix of the same shape as rmat.
- seff_var_mat: estimated spatial effect variance matrix of the same shape as rmat.
- idx: a list of two entries:
  - idx.allmissing: index of the completely missing images.
  - idx.imputed: index of the partially observed images, where spatial effects are estimated.

smooth_spline  Smoothing spline regression

Description

Smoothing spline regression

Usage

smooth_spline(x, y, x.eval = x, minimum.num.obs = 4, ...)

Arguments

x  independent variable
y  response variable
x.eval  vector to predict on
minimum.num.obs  minimum number of observations needed to run the regression
...  other parameters to be passed to smooth.spline function

Value

predicted values at 'x.eval'
spreg

spline regression

Description

spline regression

Usage

spreg(
  x,
  y,
  x.eval,
  minimum.num.obs = 4,
  basis = c("fourier", "bspline"),
  rangeval = c(min(x.eval) - 1, max(x.eval)),
  nbasis = 11,
  ...
)

Arguments

x          independent variable
y          response variable
x.eval     vector to predict on
minimum.num.obs
            minimum number of observations needed to run the regression
basis      what basis to use, "fourier" and "bspline" are available
rangeval  see fda::create.basis
nbasis     see fda::create.basis
...        arguments passed to fda::create.basis functions

Value

predicted values at ‘x.eval’
stfit_landsat

STFIT for Landsat data

Description

This function is used for Landsat data imputation, which includes five steps: mean estimation, outlier detection, temporal effect estimation, spatial effect estimation and imputation. In real application, one can use this as a template to create a five steps imputation procedure depending on the real data structure.

Usage

stfit_landsat(
  year,
  doy,
  mat,
  img.nrow,
  img.ncol,
  doyeval = 1:365,
  h.tcov = 100,
  h.tsigma2 = 300,
  h.scov = 2,
  h.ssigma2 = 2,
  nnr = 10,
  outlier.action = c("keep", "remove"),
  outlier.tol = 0.2,
  intermediate.save = TRUE,
  intermediate.dir = ".intermediate_output/",
  use.intermediate.result = TRUE,
  teff = TRUE,
  seff = TRUE,
  doy.break = NULL,
  cycle = FALSE,
  t.grid = NULL,
  t.grid.num = 50,
  clipRange = c(0, 1800),
  clipMethod = "nnr",
  var.est = FALSE
)

Arguments

year  vecotr of year

doy  vecotr of DOY (day of the year)

mat  a numeric matrix. Each row contains a row stacked image pixel values.

img.nrow  number of rows of the image
stfit_landsat

**img.ncol**
number of columns of the image

**doyeval**
a vector of DOY on which to get the mean and temporal imputation

**h.tcov**
bandwidth for temporal covariance estimation

**h.tsigma2**
bandwidth for temporal variance estimation

**h.scov**
bandwidth for spatial covariance estimation

**h.ssigma2**
bandwidth for spatial variance estimation

**nrr**
maximum number of nearest neighbor pixels to use for spatial covariance estimation

**outlier.action**
"keep" to keep outliers; "remove" to replace outliers with imputed values

**outlier.tol**
The threshold to use to define outlier image. Default is 0.2, i.e. images with more than 20% outlier pixels are treated as outlier image.

**intermediate.save**
TRUE or FALSE; whether to save the intermediate results including mean, temporal effect and spacial effect imputation results. The intermediate results can be useful to avoid duplicating the computation for some imputation steps.

**intermediate.dir**
directory where to save the intermediate results

**use.intermediate.result**
whether to use the intermediate results in the 'intermediate.dir' folder. Default is TRUE.

**teff**
TRUE or FALSE, wheter to calculate the temporal effect. Default is TRUE.

**seff**
TRUE or FALSE, wheter to calculate the spatial effect. Default is TRUE.

**doy.break**
a vector of break points for doy where the spatial effect are estimated seperately on each interval. Default is NULL, i.e. the spatial effect is assumed to be the same over doy.

**cycle**
TRUE or FALSE. When doy.break is specified, whether to combine the first doy.break interval and the last doy.break together for spatial effect estimation.

**t.grid**
a vector of grid points on which to calculate the temporal covariance function

**t.grid.num**
number of grid points to use for temporal covariance estimation. Ignored if t.grid is given.

**clipRange**
passed to meanEst function

**clipMethod**
passed to meanEst function

**var.est**
Whether to estimate the variance of the temporal and spatial effects. Default is FALSE.

**Value**
List of length 4 with entries:

- imat: imputed matrix of mat
- smat: standard error matrix of the same size as mat
- idx: a list of image indexes
- idx.allmissing: completely missing image indexes,
- idx.partialmissing: partially observed image indexes,
- idx.fullyobserved: fully observed image indexes,
- idx.outlier: outlier image indexes.

• outlier: a list of image outliers information
  - outidx: image index with outlier pixels,
  - outpct: percentage of outlier pixels corresponding to outidx,
  - outlst: a list of the same length as outidx, with each list the missing pixel index.

Examples

library(doParallel)
library(raster)
library(rasterVis)
library(RColorBrewer)
dFB = landsat106[landsat106$year >= 2000,]
matB = as.matrix(dFB[-c(1:2)])
year = dFB$year
doy = dFB$doy
if(require(doParallel))
  registerDoParallel(1)
res <- stfit_landsat(year, doy, matB, 31, 31, nnr=30,
                     use.intermediate.result = FALSE, intermediate.save = FALSE, var.est = TRUE)
## visualize the imputed results
idx = c(res$idx$idx.allmissing[150], res$idx$idx.partialmissing[c(30, 60, 90)])
rst_list = list()
for(i in 1:length(idx)){
  rst_list[(i-1)*3+1] = raster(matrix(matB[idx[i],], 31))
  rst_list[(i-1)*3+2] = raster(matrix(res$imat[idx[i],], 31))
  rst_list[(i-1)*3+3] = raster(matrix(res$sdmat[idx[i],], 31))
}
s = stack(rst_list)
levelplot(s, index.cond=list(c(seq(1, 12, 3), seq(2, 12, 3), seq(3, 12, 3))),
          par.setting = rasterTheme(panel.background=list(col="black"),
                                    region = brewer.pal(9, 'YlOrRd'),
                                    names.attr = c(rbind(paste0("Original ", idx),
                                   paste0("Imputed ", idx),
                                   paste0("Std. Error ", idx))),
          layout = c(4,3))

---

**Description**

STFIT Temporal Effect Estimation
teffEst

Usage

```r
teffEst(
  ids,
  doy,
  rmat,
  doyeval = seq(min(doy), max(doy)),
  h.cov = 100,
  h.sigma2 = 300,
  weight.cov = NULL,
  weight.sigma2 = NULL,
  pve = 0.99,
  t.grid = NULL,
  t.grid.num = 50,
  var.est = FALSE
)
```

Arguments

- `ids`: ids for 'group', for data with repeated measurement over years, year is ids; for pixels belong to certain clusters, cluster is ids.
- `doy`: vector of DOY (day of the year)
- `rmat`: residual matrix with rows corresponding to doy and columns corresponding to pixel index
- `doyeval`: a vector of DOY on which to get the temporal imputation
- `h.cov`: bandwidth for temporal covariance estimation; ignored if `weight.cov` is supplied
- `h.sigma2`: bandwidth for temporal variance estimation
- `weight.cov`: weight vector for temporal covariance estimation
- `weight.sigma2`: weight vector for temporal variance estimation
- `pve`: percentage of variance explained; used for number of eigen values selection. Default is 0.99.
- `t.grid`: a vector of grid points on which to calculate the temporal covariance function
- `t.grid.num`: number of grid points to use for temporal covariance estimation. Ignored if `t.grid` is given.
- `var.est`: Whether to estimate the variance of the temporal effect. Default is FALSE.

Value

List of length 2 with entries:

- `teff_array`: 3-d array with first dimension 'ids', second dimension 'doy' and third dimension pixel index.
- `teff_var_array`: same structure as `teff_array` if `var.est` is TRUE, otherwise NULL.
weightMatrix  

**Description**
Weight matrix calculation

**Usage**
weightMatrix(h)

**Arguments**
- h 'bandwith'

**Value**
a weighting matrix

---

weightVector  

**Description**
Weight vector calculation

**Usage**
weightVector(h)

**Arguments**
- h bandwidth, should be positive numbers

**Value**
a vector
Index

* datasets
  landsat106, 4
  opts_stfit, 10

ARE, 2
epan, 3
getMask, 3
getMissingLayers, 4

landsat106, 4
landsat2 (landsat106), 4
landsatVis, 5
lc_cov_1d, 5
lc_cov_1d_est, 6
llreg, 6
lpreg, 7

meanEst, 8
NMSE, 10
opts_stfit, 10
outlier, 11
pctMissing, 11
rmOutlier, 12
RMSE, 12

seffEst, 13
smooth_spline, 14
spreg, 15
stfit-package, 2
stfit_landsat, 16

teffEst, 18

weightMatrix, 20
weightVector, 20