Package 'sephora'

January 17, 2024

Version 0.1.31

Date 2024-01-16

Title Statistical Estimation of Phenological Parameters

Description Provides functions and methods for estimating phenological dates (green up, start of a season, maturity, senescence, end of a season and dormancy) from (nearly) periodic Earth Observation time series. These dates are critical points of some derivatives of an idealized curve which, in turn, is obtained through a functional principal component analysis-based regression model. Some of the methods implemented here are based on T. Krivobokova, P. Serra and F. Rosales (2022) <https://www.sciencedirect.com/science/article/pii/S0167947322000998>. Methods for handling and plotting Earth observation time series are also provided.

LazyData yes

License GPL (>= 2)

Encoding UTF-8

Depends R (>= 3.5.0), ggplot2 (>= 3.3.6), methods, geoTS (>= 0.1.7), eBsc (>= 4.15), rootSolve (>= 1.8.2.3)

Imports dtwclust (>= 5.5.10), foreach (>= 1.4.4), parallel (>= 3.6.1), doParallel (>= 1.0.14), dplyr (>= 1.0.9), nlme (>= 3.1-157), MASS (>= 7.3-57), ggnewscale (>= 0.4.7), spiralize (>= 1.0.6)

Suggests TSclust (>= 1.3.1), bigmemory (>= 4.6.1), vcd (>= 1.4-11)

NeedsCompilation no

RoxygenNote 7.2.3

Collate 'auxFUN.R' 'data.R' 'datesToDoY.R' 'fill_initialgap_MOD13Q1.R' 'getDist_phenoParam.R' 'getSpiralPlot.R' 'get_metadata_years.R' 'global_min_max.R' 'local_min_max.R' 'ndvi_derivatives.R' 'phenopar.R' 'phenopar_polygon.R' 'plot.R' 'sephora-class.R' 'sephora-methods.R' 'sephora-package.R' 'vecFromData.R' 'vecToMatrix.R'

Author Inder Tecuapetla-Gómez [cre, aut] (0000-0001-6251-972X), Fanny Galicia-Gómez [ctb], Francisco Rosales-Marticorena [ctb]

Maintainer Inder Tecuapetla-Gómez <itecuapetla@conabio.gob.mx>

Repository CRAN

Date/Publication 2024-01-17 18:40:02 UTC

R topics documented:

`sephora-package` *Statistical Estimation of Phenological Parameters*

Description

Estimates phenological dates of satellite imagery time series. Originally conceived to handle MODIS time series (especifically MOD13Q1), this package can handle Earth Observation time series from any satellite mission.

Details

The main function of this package, `phenopar`, allows a `numeric` vector containing satellite-based measurements (preferably, vegetation indices for better results). These observations can be construed as realizations of an underlying periodic stochastic process that has been recorded from the first day of the year (DoY) of `startYear` to the last DoY of `endYear`. Thus, each numeric vector can be assembled as a matrix whose number of rows and columns equal to `length(startYear:endYear)` and `frequency`, respectively, see `get_metadata_years`. Moreover, each row of this matrix can be thought as the realization of the periodic stochastic process throughout a season. Thus, having multiple measurements of such a process, functional principal component methods are employed to extract an underlying idealized (vegetation index) curve.

The phenological dates that can be estimated with sephora are:

- **Green Up (GU)**.

- **Start of Season (SoS)**.
- **Maturity (Mat)**.
- **Senescence (Sen)**.
- **End of Season (EoS)**.
- **Dormancy (Dor)**.

Data handling

The following functions allow to access numeric vectors of time series satellite imagery, in particular, MOD13Q1 time series starting at February 18, 2000.

`fill_initialgap_MOD13Q1`	Fill first 3 MOD13Q1 observations
`vecFromData`	Get numeric vector from an RData file
`vecToMatrix`	Set numeric vector as a matrix
`get_metadata_years`	Get metadata useful in certain visualizations

Modeling

The following functions allow to smooth out and fit a regression model based on Functional Principal Components. Applications of these functions allow to estimate phenological parameters of numeric vectors of Earth Observation time series:

`ndvi_derivatives`	Derivatives of idealized NDVI curve
`phenopar`	Estimate phenological dates
`phenopar_polygon`	Estimate phenological dates (parallel processing)

Plotting

Plot methods for numeric and `sephora` objects:

`getSpiralPlot`	Spiral plot of polygon-based phenological date estimates
`plot.sephora`	Plot methods for `sephora-class` object

Miscellaneous

`datesToDoY`	Maps estimated phenological dates to days of a year
`getDist_phenoParam`	Access to vectors of phenological date estimates from a list
`global_min_max`	Global critical points of a curve on a closed interval
`local_min_max`	Local critical points of a curve on a union of open intervals

Author(s)

Tecuapetla-Gómez, I. <itecuapetla@conabio.gob.mx>

`datesToDoY` *Mapping phenodates to days of year (DoY)*

Description

This function maps estimated phenological dates to days of a year.

Usage

```
datesToDoY(
  start = 1,
  end = 12,
  phenodates,
  totalDoY = c(0, cumsum(c(31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31)))
)
```

Arguments

`start`	numeric, first month in mapping range. Default is 1.
`end`	numeric, last month in mapping range. Default is 12.
`phenodates`	numeric vector of length 6 containing estimates of phenological dates (green up, start of season, maturity, senescence, end of season and dormancy)
`totalDoY`	numeric vector, each entry (except for the first) gives a month's total number of days

Details

Length of `start:end` must be equal to `length(totalDoY)-1`.

Value

A `data.frame` with variables `month` and `day`

Examples

```
x <- c(102,140,177,301,339,242)
names(x) <- c("GU", "SoS", "Mat", "Sen", "EoS", "Dor")
datesToDoY(phenodates = x)
```

deciduous_polygon *128 NDVI pixels from a MOD13Q1 time series*

Description

Small spatial subset of a MOD13Q1 time series from 2000 to 2021. The MOD13Q1 provides measurements of the Normalized Difference Vegetation Index (NDVI), a variable that is suitable to conduct mid-term vegetation studies remotely. The pixels provided by this dataset were recorded from a deciduous forest zone.

Usage

```
data(deciduous_polygon)
```

Format

An object of class `matrix`.

Details

The dataset is distributed through an `RData` file containing a `matrix` object with 128 rows and 506 columns.

fill_initialgap_MOD13Q1 *Fill gaps of first three dates of MOD13Q1*

Description

Since MOD13Q1 was released on 18-02-2000 and its temporal resolution is 16 days, there are no measurements available for the first three acquisition dates of 2000. This function allows to fill these three dates using historic data.

Usage

```
fill_initialgap_MOD13Q1(m, fun = stats::median)
```

Arguments

`m`	matrix with `nrow` equal to the number of periods (seasons or years) studied, and `ncol` equal to the number of observations per period.
`fun`	a function employed to impute missing values. Default, `stats::median`.

Details

The missing values of `m` are `m[1,1]`, `m[1,2]` and `m[1,3]`. For instance, to fill `m[1,1]` the values of `m[2:nrow(m),1]` are used, and consequently, it is expected that the larger the numeric vector, the smaller the variability of the imputed value for `m[1,1]`.

Value

A numeric vector of length 3

Note

It is recommended to use `vecToMatrix` to transfer the values of a numeric vector of MOD13Q1 measurements into a matrix.

See Also

`vecToMatrix`, `vecFromData`

Examples

```
data("deciduous_polygon")
str(deciduous_polygon, vec.len = 1)
x <- deciduous_polygon[1,] # check x[1:3]
x_asMatrix <- vecToMatrix(x, lenPeriod = 23) # check str(x_asMatrix)
x_asMat_complete <- fill_initialgap_MOD13Q1(m=x_asMatrix)

#filled first three values of x
x[1:3] <- x_asMat_complete
```

---

getDist_phenoParam *Utility function*

Description

Extracts an estimated phenological parameter from a `list`. Useful when `phenopar_polygon` was applied to estimate phenological dates over a polygon.

Usage

```
getDist_phenoParam(
  LIST,
  phenoParam = c("GU", "SoS", "Mat", "Sen", "EoS", "Dor")
)
```

Arguments

`LIST`	list, containing 6 estimated phenological parameters
`phenoParam`	character. What phenological parameter should be extracted?

Value

A numeric vector

See Also

getSpiralPlot, phenopar_polygon

`getSpiralPlot` *Spiral plot of phenological parameters*

Description

This utility function yields a spiral plot based on phenological dates estimated from a polygon.

Usage

```
getSpiralPlot(LIST, MAT = NULL, height = 0.2, LABELS, ...)
```

Arguments

`LIST`	list, containing 6 estimated phenological parameters.
`MAT`	matrix, containing 6 estimated phenological parameters. Default, `NULL`.
`height`	numeric, height parameter of `spiral_track` (used internally)
`LABELS`	character, labels parameter of `spiral_axis` (used internally)
`...`	additional parameters to `spiral_initialize`

Value

No value is returned

See Also

getSpiralPlot, phenopar_polygon, spiral_track, spiral_axis, spiral_initialize

get_metadata_years *Returns metadata to construct x-axis and legend of* plot.sephora

Description

Metadata either from a numeric vector or a sephora-class object

Usage

```
get_metadata_years(x, startYear = 2000, endYear = 2021, frequency = 23)
```

Arguments

x	numeric vector or sephora-class object
startYear	integer, x initial year
endYear	integer, x final year
frequency	integer giving number of observations per season. Default is 23.

Value

A list of 2 components:

xDates	date vector containing DoY (acquisition date) using format yyyy-mm-dd
xLabels	character vector containing period of study years using format "'YY"

Examples

```
x <- deciduous_polygon[1,]
y <- get_metadata_years(x=x)
str(y)
```

global_min_max *Global minimum and maximum of a real-valued continuous function over a closed interval*

Description

Gets global minimum and maximum of a given function expression on an interval using basic calculus criteria

Usage

```
global_min_max(f, f1der, f2der, D)
```

Arguments

f	function expression
f1der	function expression of first derivative of f
f2der	function expression of second derivative of f
D	numeric vector specifying the interval over which f is optimized

Details

This function uses uniroot.all to get all roots of f1der over D, additionally, the second derivative criterion is used to determine the global minimum and maximum.

Value

A list containing:

min	numeric giving critical point where global minimum is achieved
max	numeric giving critical point where global maximum is achieved
mins	numeric vector giving all critical points satisfying second derivative criterion for minimum
maxs	numeric vector giving all critical points satisfying second derivative criterion for maximum

See Also

phenopar, uniroot.all

---

local_min_max *Local minimum and maximum of a real-valued continuous function over an open interval*

Description

Gets local minimum and maximum of a given function expression on an interval using basic calculus criteria

Usage

```
local_min_max(f, f1der, f2der, what = c("min", "max"), x0, D)
```

Arguments

f	function expression
f1der	function expression of first derivative of f
f2der	function expression of second derivative of f
what	character. What to look for? A local min or a max?
x0	numeric givin global minimum or maximum of f over the the interval D.
D	numeric vector specifying the interval over which f is optimized

Details

This function looks for critical values over the interval `[D[1],x0-1)` ∪ `(x0+1, D[length(D)]]`.

Value

A list containing:

- `x_opt` numeric giving the critical point where the local min or max is achieved. When local min or max cannot be determined, this function returns `NA`.
- `locals` numeric vector giving all critical points satisfying second derivative criteria.
- `crtPts` a list with 2 entries:
  - `x_d1` numeric vector with local critical points over `[D[1],x-1)`
  - `x_d2` numeric vector with local critical points over `(x0+1,D[length(D)]]`
- `type` character, what was found? A `min` or a `max`?

See Also

`global_min_max`, `phenopar`

`ndvi_derivatives` *Calculates derivatives of idealized NDVI*

Description

Provides function expression of derivatives of an idealized NDVI curve fitted through a harmonic regression model

Usage

```
ndvi_derivatives(amp, pha, degree, L)
```

Arguments

`amp`	numeric vector specifying amplitude parameter
`pha`	numeric vector specifying phase angle parameter
`degree`	integer. What derivative's degree should be calculated? `degree=0` corresponds to harmonic regression fit
`L`	integer giving the number of observations per period

Details

This function returns the derivatives of $f(t)$, with respect to t, when f has the representation:

$f(t) = \sum_{k=1}^{p} a[i]cos((2\pi kt)/L - \phi[i])$,

where a and ϕ are substituted by the vectors `amp` and `phase`, respectively. The degree of the derivative is given by the argument `degree`.

Value

A function expression

Note

For historic reasons, we ended up using the name `ndvi_derivatives` for this function, but it can be used to calculate derivatives of any function expression defined through `amp`, `pha`, `degree` and `L`.

See Also

`phenopar`, `phenopar_polygon`, `haRmonics`

---

phenopar *Phenological parameters estimation*

Description

Estimation of 6 phenological parameters from a numeric vector. The estimated parameters are: **green up**, **start of season**, **maturity**, **senescence**, **end of season** and **dormancy**. These parameters are critical points of some derivatives of an idealized curve which, in turn, is obtained through a functional principal component analysis (FPCA)-based regression model.

Usage

```
phenopar(
  x,
  startYear,
  endYear,
  frequency = 23,
  method = c("OLS", "WLS"),
  sigma = NULL,
  numFreq,
  delta = 0,
  distance,
  samples,
  basis,
  corr = NULL,
  k,
  trace = FALSE
)
```

Arguments

`x`	a numeric vector.
`startYear`	integer, time series initial year
`endYear`	integer, time series final year

frequency	integer giving number of observations per season. Default, 23.
method	character. Should OLS or WLS be used for smoothing x through a harmonic regression model. See **Details**.
sigma	numeric vector of length equal to frequency. Each entry gives the standard deviation of observations acquired at same day of the year. Pertinent when method=WLS only.
numFreq	integer specifying number of frequencies used in harmonic regression model.
delta	numeric. Default, 0. When harmonic regression problem is ill-posed, this parameter allows a simple regularization. See **Details**.
distance	character indicating what distance to use in hierarchical clustering. All distances in tsclust are allowed. See **Details**.
samples	integer with number of samples to draw from smoothed version of x. Used exclusively in Functional Principal Components Analysis (FPCA)-based regression. See **Details**.
basis	list giving numeric basis used in FPCA-based regression. See **Details**.
corr	Default NULL. Object defining correlation structure, can be numeric vector, matrix or function.
k	integer, number of principal components used in FPCA-based regression.
trace	logical. If TRUE, progress on the hierarchical clustering is printed on console. Default, FALSE.

Details

In order to estimate the phenological parameters, first x is assembled as a matrix. This matrix has as many rows as years (length(startYear:endYear)) in the studied period and as many columns as observations (frequency) per year. Then, each vector row is smoothed through the harmonic regression model haRmonics. This function allows for homogeneous (OLS) and heterogeneous (WLS) errors in the model. When method=WLS, sigma must be provided, hetervar is recommended for such a purpose. Additional parameters for haRmonics are numFreq and delta.

Next, equally spaced samples are drawn from each harmonic regression fit, the resulting observations are stored in the matrix m_aug_smooth. tsclust is applied to m_aug_smooth in order to obtain clusters of years sharing similar characteristics; 2 clusters are produced. The next step is applied to the dominating cluster (the one with the majority of years, >=10), or to the whole of columns of m_aug_smooth when no dominating cluster can be determined.

Based on the observations produced in the hierarchical clustering step, a regression model with the following representation is applied:

$f_i(t) = \tau(t) + \sum_{j=1}^{k} \varepsilon_j(t)\nu_{ij} + \epsilon_i,$

where $f_i(t)$ is substituted by the vector of sample observations of the i-th year; $\varepsilon_j(t)$ is the j-th functional principal component (FPC); ν_{ij} is the score associated with the j-th FPC and the i-th vector of sampled observations; and ϵ_i is a normally distributed random variable with variance σ^2, see *Krivobokova et al. (2022)* for further details. From this step, an estimate of τ is produced -fpca- this is an idealized version of the original observations contained in x.

Parameter basis can be supplied through a call to drbasis with parameters nn=samples and qq=2. Parameter corr indicates whether correlation between annual curves must be considered; the current implementation does not incorporate correlation. The number of principal components is controlled by k.

Next, a harmonic regression is fitted to fpca (a numeric vector of length equal to samples) with the parameters provided above (method, sigma, numFreq, delta). Based on the estimated parameters of this fit (fpca_harmfit_params) a R function is calculated along with its first, second, third and fourth derivatives. These derivatives are used in establishing the phenological parameters (phenoparams) utilizing basic calculus criteria similar to what *Baumann et al. (2017)* have proposed.

Finally, when 6 phenoparams are found status=Success, otherwise status=Partial.

Value

A sephora-class object containing 14 elements

x	numeric vector
startYear	integer, time series initial year
endYear	integer, time series final year
freq	numeric giving number of observations per season. Default is 23.
sigma	when method="OLS", numeric of length one (standard deviation); when method="WLS", numeric vector of length equal to freq
m_aug_smooth	matrix with nrow=samples and ncol=(length(x)/freq) containing sampled observations
clustering	Formal class HierarchicalTSClusters with 20 slots. Output from a call to tsclust with parameters series=m_aug_smooth, type='h', distance=distance
fpca	numeric vector of length equal to samples
fpca_harmfit_params	list of 4: a.coef, b.coef, amplitude and phase as in haRmonics output.
fpca_fun_0der	function, harmonic fit for x
fpca_fun_1der	function, first derivative of harmonic fit for x
fpca_fun_2der	function, second derivative of harmonic fit for x
fpca_fun_3der	function, third derivative of harmonic fit for x
fpca_fun_4der	function, fourth derivative of harmonic fit for x
phenoparams	named numeric vector of length 6
status	character, specifying whether FPCA model was inverted successfully (Success) or partially ("Partial"). In other words, Success and Partial mean that 6 or less than 6 parameters were estimated, respectively.

References

Krivobokova, T. and Serra, P. and Rosales, F. and Klockmann, K. (2022). *Joint non-parametric estimation of mean and auto-covariances for Gaussian processes*. Computational Statistics & Data Analysis, **173**, 107519.

Baumann, M. and Ozdogan, M. and Richardson, A. and Radeloff, V. (2017). *Phenology from Landsat when data is scarce: Using MODIS and Dynamic Time-Warping to combine multi-year Landsat imagery to derive annual phenology curves*. International Journal of Applied Earth Observation and Geoinformation, **54**, 72–83

See Also

haRmonics, hetervar, tsclust, drbasis.

Examples

```
# --- Load dataset for testing
data("deciduous_polygon")

# --- Extracting first pixel of deciduous_polygon
pixel_deciduous <- vecFromData(data=deciduous_polygon, numRow=3)

# --- Following objects are used in this example
# --- for CRAN testing purposes only. In real life examples
# --- there is no need to shorten time series length

EndYear <- 2010
number_observations <- 23*11

# --- needed parameter
BASIS <- drbasis(n=50, q=2)

# --- testing phenopar
sephora_deciduous <- phenopar(x=pixel_deciduous$vec[1:number_observations],
                              startYear=2000, endYear=EndYear,
                              numFreq=3, distance="dtw2",
                              samples=50, basis=BASIS, k=3)

# --- testing ndvi_derivatives
f <- ndvi_derivatives(amp = sephora_deciduous$fpca_harmfit_params$amplitude,
                      pha = sephora_deciduous$fpca_harmfit_params$phase,
                      degree = 0, L = 365)
fprime <- ndvi_derivatives(amp = sephora_deciduous$fpca_harmfit_params$amplitude,
                           pha = sephora_deciduous$fpca_harmfit_params$phase,
                           degree = 1, L = 365)
fbiprime <- ndvi_derivatives(amp = sephora_deciduous$fpca_harmfit_params$amplitude,
                             pha = sephora_deciduous$fpca_harmfit_params$phase,
                             degree = 2, L = 365)
f3prime <- ndvi_derivatives(amp = sephora_deciduous$fpca_harmfit_params$amplitude,
                            pha = sephora_deciduous$fpca_harmfit_params$phase,
                            degree = 3, L = 365)
f4prime <- ndvi_derivatives(amp = sephora_deciduous$fpca_harmfit_params$amplitude,
                            pha = sephora_deciduous$fpca_harmfit_params$phase,
                            degree = 4, L = 365)

# --- testing global_min_max and local_min_max
intervalo <- seq(1,365, length=365)
```

```
GU_Mat <- global_min_max(f=fbiprime, f1der=f3prime, f2der=f4prime, D=intervalo)
Sen <- local_min_max(f=fbiprime, f1der=f3prime, f2der=f4prime,
                     what="min", x0=GU_Mat$min, D=intervalo)
SoS_EoS <- global_min_max(f=fprime, f1der=fbiprime, f2der=f3prime, D=intervalo)
Dor <- local_min_max(f=fbiprime, f1der=f3prime, f2der=f4prime,
                     what="max", x0=GU_Mat$max, D=intervalo)

# --- phenological dates (rough estimates)
c(GU=GU_Mat$max, SoS=SoS_EoS$max, Mat=GU_Mat$min,
  Sen=Sen$x_opt, EoS=SoS_EoS$min, Dor=Dor$x_opt)
# --- phenological dates provided by sephora
sephora_deciduous$phenoparams

# --- testing plotting methods
plot(x=sephora_deciduous, yLab="NDVI (no rescaled)")
plot(x=sephora_deciduous, type="profiles",
     xLab="DoY", yLab="NDVI (no rescaled)")

# --- 2015 forms Cluster 2
plot(x=sephora_deciduous, type="ms")

# --- graphical definition of phenological dates
plot(x=sephora_deciduous, type="derivatives")

# --- Overlapping FPCA fit to original time series
gg <- plot(x=sephora_deciduous, type="profiles",
           xLab="DoY", yLab="NDVI (no rescaled)")
x_axis <- get_metadata_years(x=pixel_deciduous$vec,
                             startYear=2000, endYear=EndYear, frequency=23)
DoY <- seq(1,365, by=16)
fpca_DoY <- sephora_deciduous$fpca_fun_0der(t=DoY)
COLORS <- unique( ggplot_build(gg)$data[1][[1]]$colour )
df <- data.frame(values=c(sephora_deciduous$x, fpca_DoY),
                 years=as.factor(rep(c(x_axis$xLabels,"FPCA"), each=23)),
             DoY=factor(DoY, levels=DoY), class=c(rep(1,number_observations), rep(2,23)))
gg_fpca <- ggplot(data=df,
                  aes(x=DoY, y=values, group=years, colour=years)) +
ggplot2::geom_line(linewidth = c(rep(1,number_observations), rep(4,23))) +
ggplot2::labs(y="NDVI", x="DoY", color="years+FPCA") +
ggplot2::scale_color_manual(values = c(COLORS, "#FF4500")) +
ggplot2::theme(legend.position = "right")
gg_fpca
```

---

phenopar_polygon *Phenological parameters estimation in mass*

Description

Estimation of phenological parameters from a set of numeric vectors stored in a RData file. Output is saved as a RData file at the destination specified by `dirToSave`

Usage

```
phenopar_polygon(
  path = NULL,
  product = c("MOD13Q1", "independent"),
  data,
  frequency = 23,
  method = c("OLS", "WLS"),
  sigma = NULL,
  numFreq,
  delta = 0,
  distance,
  samples,
  basis,
  corr = NULL,
  k,
  trace = FALSE,
  numCores = 20,
  dirToSave,
  reportFileBaseName = "phenopar_progress",
  outputFileBaseName = "polygon"
)
```

Arguments

path	character with full path of RData file containing numeric vectors to analyze.
product	character specifying whether dataset is the MOD13Q1 product (default) or a different one (independent).
data	matrix with dataset to analyze. Pertinent when product="independent" only.
frequency	integer giving number of observations per season. Default, 23.
method	character. Should OLS or WLS be used for smoothing each numeric vector in RData file specified in path?
sigma	numeric vector of length equal to frequency. Each entry gives the standard deviation of observations acquired at same day of the year. Pertinent when method=WLS.
numFreq	integer specifying number of frequencies to use in harmonic regression model.
delta	numeric. Default, 0. When regression problem is ill-posed, this parameter allows a simple regularization.
distance	character indicating what distance to use in hierarchical clustering. All distances in tsclust are allowed.
samples	integer with number of samples to draw from smoothed version of numeric vector to analyze. Used exclusively in Functional Principal Components Analysis (FPCA)-based regression.
basis	list giving numeric basis used in FPCA-based regression. See details.
corr	Default NULL. Object defining correlation structure, can be numeric vector, matrix or function.

k integer, number of principal components used in FPCA-based regression.

trace logical. If TRUE, progress on the hierarchical clustering is printed on console. Default, FALSE.

numCores integer. How many processing cores can be used?

dirToSave character. In which directory to save analysis results?

reportFileBaseName
character. What base name should be given to a progress report file? Default, phenopar_progress.

outputFileBaseName
character. What base name should be given to the output file? Default, polygon.

Value

At the location specified by dirToSave, a file containing a matrix with nrow equal to the number of numeric vectors analyzed and 6 columns, is saved. The name of this file is:

paste0(tools::file_path_sans_ext(basename(path)), "_phenoparams.RData").

See Also

phenopar, getSpiralPlot, tsclust.

Examples

```
dirOUTPUT <- system.file("data", package = "sephora")
BASIS <- drbasis(n=100, q=2)

polygon_deciduous <- deciduous_polygon
for(i in 1:nrow(polygon_deciduous)){
  polygon_deciduous[i,] <- vecFromData(data=deciduous_polygon, numRow=i)$vec
}

# --- In the following example 'numCores=2' for CRAN
# --- testing purposes only. In a real life example
# --- users are encouraged to set 'numCores' to a number
# --- that reflects the size of their data set as well
# --- as the number of available cores

phenopar_polygon(data=polygon_deciduous,
                 product="independent",
                 numFreq = 3, distance = "dtw2",
                 samples=100, basis=BASIS,
                 k=3, numCores=2,
                 dirToSave=dirOUTPUT,
                 outputFileBaseName = "deciduous")

# --- Auxiliary function to read phenopar_polygon output,
# --- used below to define deciduous_params object
LoadToEnvironment <- function(RData, env = new.env()){
                              load(RData, env)
```

```
                                return(env)}

# --- colors used in spiralPlot below
cgu <- rgb(173/255,221/255,142/255)
csos <- rgb(120/255,198/255,121/255)
cmat <- rgb(49/255, 163/255,84/255)
csen <- rgb(217/255, 95/255, 14/255)
ceos <- rgb(254/255, 153/255, 41/255)
cdor <- rgb(208/255, 209/255, 230/255)

colores <- c(cgu,csos,cmat,csen,ceos,cdor)

# --- how to get a SpiralPlot
listRDatas <- list.files(path=dirOUTPUT,
                         pattern=".RData",
                         full.names=TRUE)

deciduous_params <- LoadToEnvironment(listRDatas[1])

getSpiralPlot(MAT=deciduous_params$output,
              LABELS=month.name,
              vp_param=list(width=0.5, height=0.7))
vcd::grid_legend(x=1.215, y=0.125, pch=18, col=colores,
                 frame=FALSE,
                 labels=c("GU","SoS","Mat","Sen","EoS","Dor"),
                 title="Params")

# --- cleaning up after work
unlink(paste0(dirOUTPUT, "/deciduous_phenoParams.RData"))
unlink(paste0(dirOUTPUT, "/phenopar_progress.txt"))
```

`plot.sephora` *Plot methods for* `sephora`

Description

Methods associated with `sephora-class`.

Usage

```
## S3 method for class 'sephora'
plot(
  x,
  y,
  startYear,
  endYear,
  frequency,
  type = NULL,
```

```
  sizeLine = 1,
  sizePoint = 2,
  position_legend = "none",
  title_legend = NULL,
  xLab = "Time",
  yLab = "Index",
  xLim,
  msTitle = "Cluster",
  pointShape = 16,
  pointSize = 2,
  pointStroke = 3,
  textFontface = 2,
  textSize = 5,
  text_hjust = 0.5,
  text_vjust = -0.5,
  ...
)
```

Arguments

x	a numeric vector or a sephora object.
y	ignored.
startYear	integer, time series initial year.
endYear	integer, time series final year.
frequency	integer giving number of observations per season.
type	character specifying type of plot. By default, NULL; "profiles", "ms" and "derivatives" are also allowed. See **Details.**
sizeLine	integer giving line size
sizePoint	integer giving point size
position_legend	character. Should a legend be added? Where? See theme.
title_legend	character. Should a legend be added? What would it be? See theme and **Details**.
xLab	character, label to display in x-axis.
yLab	character, label to display in y-axis. See **Details**.
xLim	date vector of length 2 indicating limits of x-axis. When no supplied, x will be displayed in the period of time defined by startYear, endYear and frequency.
msTitle	character. Default "Cluster". See **Details**.
pointShape	shape parameter used in geom_point. Default 16. See **Details**.
pointSize	size parameter used in geom_point. Default 2. See **Details**.
pointStroke	stroke parameter used in geom_point. Default 3. See **Details**.
textFontface	fontface parameter used in geom_text. Default 2. See **Details**.
textSize	size parameters used in geom_text. Default 5. See **Details**.
text_hjust	hjust parameter used in geom_text. Default 0.5. See **Details**.
text_vjust	vjust parameter used in geom_text. Default -0.5. See **Details**.
...	additional ggplot parameters.

Details

By default, type=NULL and this option allows for plotting numeric vectors and sephora objects; argument title_legend is only pertinent in this case. Other allowed options for type are "profiles", "ms" and "derivatives". When type="profiles" all the arguments used in the default case are allowed except for title_legend. When type="ms", arguments msTitle, pointShape, pointSize, pointStroke, textFontface, textSize, text_hjust and text_vjust are pertinent. When type="derivatives", the default value of argument yLab will be used.

Value

A gg object (or NULL (invisible) when type="derivatives").

Plotting

This function draws either a graphic based on a ggplot or a plot object.

The default is intended for numeric vectors and sephora-class objects. This method employs the ggplot2 system and returns a sort of time series plot.

The method *profiles*, selected when type="profiles", is also intended for numeric vectors and sephora-class objects. This method is based on the ggplot2 system and draws p curves, one for each period (p=length(startYear:endYear)), on the same time scale (days of the year).

The method *ms*, selected when type="ms", is intended for sephora-class objects only. Using the ggplot2 system this method draws the result of a multidimensional scaling analysis performed on the smoothed version of the p curves described above.

The method *derivative*, selected when type="derivatives", is intended for sephora-class objects only. A 5-panel plot is drawn showing (from top to bottom):

- FPCA estimate: the fpca entry of sephora-class object. See phenopar.
- First, second, third and fourht derivative of FPCA estimate: curve obtained by applying ndvi_derivatives to FPCA estimate.

---

sephora-class *class sephora*

Description

Definition of the sephora class

Slots

x Original time series (as a numeric vector)

startYear Beginning of time series

endYear End of time series

freq Number of observations per season

sigma Variability estimate

`m_aug_smooth` Samples of smoothed version of `x`, in matricial form

`clustering` An object of class `HierarchicalTSClusters`

`fpca` Numeric, FPCA-based regression fit

`fpca_harmfit_params` a list, harmonic fit

`fpca_fun_0der` Function fpca fit

`fpca_fun_1der` Function fpca fit first derivative

`fpca_fun_2der` Function fpca fit second derivative

`fpca_fun_3der` Function fpca fit third derivative

`fpca_fun_4der` Function fpca fit fourth derivative

`phenoparams` Phenological dates estimate

`status` Character, was phenopar estimation successful?

See Also

`sephora-methods`

`vecFromData` *Get numeric vector from RData file*

Description

Extract a numeric vector from an RData file

Usage

```
vecFromData(
  product = c("MOD13Q1", "independent"),
  data,
  numRow,
  lenPeriod = 23
)
```

Arguments

`product`	character indicating whether `data` comes from a `MOD13Q1` (default) time series satellite imagery or from an `independent` product.
`data`	a matrix containing measurements of subsets (polygons) of a time series of satellite images. `nrow` is equal to the number of pixels in the polygon and `ncol` is equal to the number of images in the time series.
`numRow`	numeric, number of row to extract from `data`.
`lenPeriod`	numeric, number of observations per period. Default, 23.

Details

Although the first available MOD13Q1 product dates back to 18-02-2000, when `product="MOD13Q1"` this function assumes that `data` contains observations from 01-01-2000 and `fill_initialgap_MOD13Q1` is used to impute the first three missing values of 2000.

Value

A list with two components:

`mat`	extracted vector in matricial form
`vec`	extracted vector

See Also

`fill_initialgap_MOD13Q1`, `phenopar`, `raster_intersect_sp`, `vecToMatrix`.

`vecToMatrix` *Mapping numeric vector to a matrix*

Description

Maps a vector (pixel of a satellite time series) to a matrix.

Usage

```
vecToMatrix(x, lenPeriod = 23)
```

Arguments

`x`	a numeric vector whose length must be a multiple of `lenPeriod`
`lenPeriod`	a numeric, number of observations per period

Value

A matrix with `nrow` equal to `length(x)/lenPeriod` and `ncol` equal to `lenPeriod`.

See Also

`fill_initialgap_MOD13Q1`, `phenopar`, `vecFromData`.

Index