Package 'satellite'

October 14, 2022

Title Handling and Manipulating Remote Sensing Data

Version 1.0.4

Author Thomas Nauss, Hanna Meyer, Tim Appelhans, Florian Detsch

Maintainer Florian Detsch <fdetsch@web.de>

Description Herein, we provide a broad variety of functions which are useful for handling, manipulating, and visualizing satellite-based remote sensing data. These operations range from mere data import and layer handling (eg subsetting), over Raster* typical data wrangling (eg crop, extend), to more sophisticated (pre-)processing tasks typically applied to satellite imagery (eg atmospheric and topographic correction). This functionality is complemented by a full access to the satellite layers' metadata at any stage and the documentation of performed actions in a separate log file. Currently available sensors include Landsat 4-5 (TM), 7 (ETM+), and 8 (OLI/TIRS Combined), and additional compatibility is ensured for the Landsat Global Land Survey data set.

Depends R (>= 2.10), raster, methods, utils, stats, grDevices, graphics

Imports plyr, Rcpp (>= 0.10.3), terra, tools, stats4

License MIT + file LICENSE

Suggests devtools, knitr, rgdal, testthat, rmarkdown

LazyData true

VignetteBuilder knitr

LinkingTo Rcpp

RoxygenNote 7.1.2

Encoding UTF-8

NeedsCompilation yes

Repository CRAN

Date/Publication 2021-10-12 09:10:02 UTC

R topics documented:

satellite-package *Smorgasboard for remote sensing functions.*

Description

Smorgasbord for remote sensing functions

Details

The package provides a variety of functions which are useful for handling, manipulating and visualizing remote sensing data.

Author(s)

Thomas Nauss, Hanna Meyer, Florian Detsch, Tim Appelhans

Maintainer: Florian Detsch <fdetsch@web.de>

References

Some functions are taken and/or adopted from Sarah C. Goslee (2011). Analyzing Remote Sensing Data in R: The landsat Package. Journal of Statistical Software, 43(4), 1-25, doi: 10.18637/jss.v043.i04.

alignGeometry *Align raster geometry between two data sets*

Description

Align raster data by bringing it in the same geometry and extent. If the data set is not in the same projection as the template, the alignment will be computed by reprojection. If the data has already the same projection, the data set will be cropped and aggregated prior to resampling in order to reduce computation time.

Usage

```
## S4 method for signature 'Satellite'
alignGeometry(x, template, band_codes, type, method = c("bilinear", "ngb"))

## S4 method for signature 'RasterStack'
alignGeometry(x, template, method = c("bilinear", "ngb"))

## S4 method for signature 'RasterLayer'
alignGeometry(x, template, method = c("bilinear", "ngb"))
```

Arguments

x	Satellite or Raster* object to be resampled.
template	Raster* or spatial data set from which geometry can be extracted.
band_codes	Band ID(s) to be resampled. If not supplied and type is not given, too, all bands will be considered for resampling.
type	Type of bands (e.g. VIS, NIR) which should be considered. If not supplied, all types will be processed depending and bands to be processed can be defined by band_codes.
method	Method for resampling; "bilinear" for bilinear interpolation (default) or "ngb" for nearest neighbor interpolation. See e.g. resample, projectRaster.

Value

Satellite object with aligned geometries.

raster::RasterStack object with aligned layers

raster::RasterLayer object with aligned layer

Examples

```
path <- system.file("testdata/LC8", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC8*.TIF"), full.names = TRUE)
sat <- satellite(files)

alignGeometry(sat, template = getSatDataLayer(sat, "B008n"),
              band_codes = "B001n")
```

---

brick *Convert selected layers of a Satellite object to a RasterBrick*

Description

Convert selected layers of a Satellite object to a RasterBrick

Usage

```
## S4 method for signature 'Satellite'
brick(x, layer = names(x), ...)
```

Arguments

x	an object of class 'Satellite'
layer	character vector (bcde codes) or integer vector (index) of the layers to be stacked
...	additional arguments passed on to brick

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)

brck <- brick(sat, c("B001n", "B002n", "B003n"))
brck
```

`calcAtmosCorr` *Atmospheric correction of remote sensing data*

Description

The function computes an atmospheric scattering correction and converts the sensors digital numbers to reflectances using

- absolute radiance correction
- DOS2: a dark object substraction model by Chavez (1996)
- DOS4: a dark object substratcion model by Moran et al. (1992)

Usage

```
## S4 method for signature 'Satellite'
calcAtmosCorr(x, model = c("DOS2", "DOS4"), esun_method = "RadRef")

## S4 method for signature 'RasterStack'
calcAtmosCorr(x, path_rad, esun, szen, model = c("DOS2", "DOS4"))

## S4 method for signature 'RasterLayer'
calcAtmosCorr(x, path_rad, esun, szen, model = c("DOS2", "DOS4"))
```

Arguments

`x`	Satellite or Raster* object providing the radiance at the sensor.
`model`	Model to be used to correct for 1% scattering (DOS2, DOS4).
`esun_method`	If x is a Satellite object, name of the method to be used to compute esun using one of `calcTOAIrradRadRef` ("RadRef"), `calcTOAIrradTable` ("Table") or `calcTOAIrradModel` ("Model").
`path_rad`	Path radiance, e.g. returned from `calcPathRadDOS`.
`esun`	Actual (i.e. non-normalized) TOA solar irradiance, e.g. returned from `calcTOAIrradRadRef`, `calcTOAIrradTable` or `calcTOAIrradModel`.
`szen`	Sun zenith angle.

Details

If a Satellite object is passed to the function, and if the required pre-processing has not been performed already, the path radiance is computed based on a dark object's scaled count value using `calcPathRadDOS` which will also take care of the TOA solar irradiance by calling `calcTOAIrradModel`, `calcTOAIrradRadRef` or `calcTOAIrradTable` (depending on `esun_method`) if necessary. The bands' scaled counts are converted to radiance using `convSC2Rad`.

The radiometric correction is based on a dark object approach using either the DOS2 (Chavez 1996) or DOS4 (Moran et al. 1992) model.

The minimum reflectance values for the dark object are identified using the approximation of Chavez (1988, see `calcPathRadDOS` for details).

The estimated values of the solar irradiance required for the path radiance can be computed by one of `calcTOAIrradTable` which is used to get readily published values of ESun, `calcTOAIrradRadRef` which computes ESun based on the actual radiance and reflectance in the scene, or `calcTOAIrradModel` which computes ESun based on look-up tables for the sensor's relative spectral response and solar irradiation spectral data.

The atmospheric transmittance towards the sensor (Tv) is approximated by 1.0 (DOS2, Chavez 1996) or Rayleigh scattering (DOS4, Moran et al. 1992).

The atmospheric transmittance from the sun (Tz) is approximated by the cosine of the sun zenith angle (DOS2, Chavez 1996) or again using Rayleigh scattering (DOS4, Moran et al. 1992).

The downwelling diffuse irradiance is approximated by 0.0 (DOS2, Chavez 1996) or the hemispherical integral of the path radiance (DOS4, Moran et al. 1992).

Equations are taken from Song et al. (2001).

Value

Satellite object with added atmospheric corrected layers

raster::RasterStack object with atmospheric corrected layers

raster::RasterLayer object with atmospheric corrected layer

References

Chavez Jr PS (1988) An improved dark-object subtraction technique for atmospheric scattering correction of multispectral data. Remote Sensing of Environment 24/3, doi: 10.1016/00344257(88)90019-3.

Chavez Jr PS (1996) Image-based atmospheric corrections revisited and improved. Photogrammetric Engineering and Remote Sensing 62/9, available online at https://www.researchgate.net/publication/236769129_Image-Based_Atmospheric_Corrections_-_Revisited_and_Improved

Goslee SC (2011) Analyzing Remote Sensing Data in R: The landsat Package. Journal of Statistical Software,43/4, 1-25, doi: 10.18637/jss.v043.i04.

Moran MS, Jackson RD, Slater PN, Teillet PM (1992) Evaluation of simplified procedures for retrieval of land surface reflectance factors from satellite sensor output.Remote Sensing of Environment 41/2-3, 169-184, doi: 10.1016/00344257(92)90076V.

Song C, Woodcock CE, Seto KC, Lenney MP, Macomber SA (2001) Classification and Change Detection Using Landsat TM Data: When and How to Correct Atmospheric Effects? Remote Sensing of Environment 75/2, doi: 10.1016/S00344257(00)001693.

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)
sat_atmos <- calcAtmosCorr(sat, model = "DOS2", esun_method = "RadRef")

bcde <- "B002n"

sat <- calcTOAIrradRadRef(sat, normalize = FALSE)

path_rad <- calcPathRadDOS(x = min(getValues(getSatDataLayer(sat, bcde))),
                           bnbr = getSatLNBR(sat, bcde),
                           band_wls =
                             data.frame(LMIN =
                                          getSatLMIN(sat,
                                                     getSatBCDESolar(sat)),
                                        LMAX =
                                          getSatLMAX(sat,
                                                     getSatBCDESolar(sat))),
                           radm = getSatRADM(sat, getSatBCDESolar(sat)),
                           rada = getSatRADA(sat, getSatBCDESolar(sat)),
                           szen = getSatSZEN(sat, getSatBCDESolar(sat)),
                           esun = getSatESUN(sat, getSatBCDESolar(sat)),
                           model = "DOS2")

sensor_rad <- convSC2Rad(x = getSatDataLayer(sat, bcde),
                         mult = getSatRADM(sat, bcde),
                         add = getSatRADA(sat, bcde), getSatSZEN(sat, bcde))

ref_atmos <- calcAtmosCorr(x = sensor_rad,
                           path_rad = path_rad[names(path_rad) == bcde],
                           esun = getSatESUN(sat, bcde),
                           szen = getSatSZEN(sat, bcde),
                           model = "DOS2")
```

`calcDODN` *Compile dark object DN for given sensor band*

Description

The function estimates the DN value of a "dark object" which is used for atmospheric correction using the DOS2 and DOS4 model. Therefore, the frequency distribution of the smallest 1% of the data values is analyzed and the value for which the first derivate has the absolute maximum is taken as the DN for a dark object.

Usage

```
## S4 method for signature 'Satellite'
```

```
calcDODN(x, bcde)

## S4 method for signature 'RasterLayer'
calcDODN(x)
```

Arguments

x	Satellite object or RasterLayer with sensor band data, e.g. returned by getSatDataLayer.
bcde	If 'x' is a Satellite object, a band code as character.

Details

The DN for a dark object is extracted from a histogram similar to Chavez (1988).

Value

Numeric value of the DN for the dark object.

References

Chavez Jr PS (1988) An improved dark-object subtraction technique for atmospheric scattering correction of multispectral data. Remote Sensing of Environment 24/3, doi: 10.1016/00344257(88)90019-3.

See Also

The DN is used by calcPathRadDOS for computing the path radiance based on the dark object method.

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)

calcDODN(sat, bcde = "B002n")
calcDODN(getSatDataLayer(sat, bcde = "B002n"))
```

---

calcEarthSunDist *Compute earth-sun distance based on day of the year*

Description

The earth-sun distance for a particular day of the year is computed based on one of several empirical formulas.

Usage

```
calcEarthSunDist(date, formula = c("Spencer", "Mather", "ESA", "Duffie"))
```

Arguments

date	Date of the sensor overpass; either a character string in a native date format (e.g. "YYYY-MM-DD", see as.Date) or a POSIX* object (see as.POSIXct).
formula	Formula to be applied, specified through the name of the author, i.e. one of "Spencer", "Mather", "ESA" or "Duffie" (see 'Details').

Details

Computation of earth-sun distance using formulas provided by Spencer (1971), Mather (2005) or ESA. If formula = "Duffie", the inverse squared relative earth–sun distance is returned as proposed by Duffie and Beckman (1980).

Value

Numeric earth-sun distance (in AU) or, if formula = "Duffie", the relative squared earth–sun distance on the given day.

References

The formulas are taken from the following sources:

- Spencer: Spencer JW (1971) Fourier series representation of the position of the sun. Search 2/5. Taken from https://goo.gl/lhi9UI.
- Mather: Mather PM (2005) Computer Processing of Remotely-Sensed Images: An Introduction. Wiley: Chichester, ISBN: 978-0-470-02101-9, https://eu.wiley.com/WileyCDA/WileyTitle/productCd-0470021012.html.
- ESA: ESA Earth Observation Quality Control: Landsat frequently asked questions.
- Duffie: Duffie JA, Beckman WA (2013) Solar Engineering of Thermal Processes. Wiley: Hoboken, New Jersey, ISBN: 978-0-470-87366-3, https://eu.wiley.com/WileyCDA/WileyTitle/productCd-0470873663.html.

See also: Bird R, Riordan C (1984) Simple solar spectral model for direct and diffuse irradiance on horizontal and tilted planes at the Earth's surface for cloudless atmospheres. Task No. 3434.10, Solar Energy Research Institute: Golden, Colorado, http://www.nrel.gov/docs/legosti/old/2436.pdf.

Examples

```
calcEarthSunDist(date = "2015-01-01", formula = "Spencer") # absolute
calcEarthSunDist(date = "2015-01-01", formula = "Duffie")  # relative
```

`calcPathRadDOS` *Compute path radiance based on the dark object method*

Description

Compute an estimated path radiance for all sensor bands, which can then be used to roughly correct the radiance values for atmospheric scattering. Path radiance estimation is based on a dark object method.

Usage

```
## S4 method for signature 'Satellite'
calcPathRadDOS(x, model = c("DOS2", "DOS4"), esun_method = "RadRef")

## S4 method for signature 'numeric'
calcPathRadDOS(
  x,
  bnbr,
  band_wls,
  radm,
  rada,
  szen,
  esun,
  model = c("DOS2", "DOS4"),
  scat_coef = c(-4, -2, -1, -0.7, -0.5),
  dos_adjust = 0.01
)
```

Arguments

`x`	A Satellite object or the value (scaled count) of a dark object in `bnbr` (e.g. minimum raw count of selected raster `bnbr`). If x is a Satellite object, the value is computed using `calcDODN`.
`model`	Model to be used to correct for 1% scattering (DOS2, DOS4; must be the same as used by `calcAtmosCorr`).
`esun_method`	If x is a Satellite object, name of the method to be used to compute esun using one of `calcTOAIrradRadRef` ("RadRef"), `calcTOAIrradTable` ("Table") or `calcTOAIrradModel` ("Model")
`bnbr`	Band number for which DNmin is valid.
`band_wls`	Band wavelengths to be corrected; `data.frame` with min (max) in first (second) column, see details.
`radm`	Multiplicative coefficient for radiance transformation (i.e. slope).
`rada`	Additive coefficient for radiance transformation (i.e. offset).
`szen`	Sun zenith angle.

esun	Actual (i.e. non-normalized) TOA solar irradiance, e.g. returned by calcTOAIrradRadRef, calcTOAIrradTable or calcTOAIrradModel.
scat_coef	Scattering coefficient; defaults to -4.0.
dos_adjust	Assumed reflection for dark object adjustment; defaults to 0.01.

Details

If x is a Satellite object, the minimum raw count value (x) is computed using calcDODN. If the TOA solar irradiance is not part of the Satellite object's metadata, it is computed using calcTOAIrradRadRef, calcTOAIrradTable or calcTOAIrradModel.

The dark object subtraction approach is based on an approximation of the atmospheric path radiance (i.e. upwelling radiation which is scattered into the sensors field of view, aka haze) using the reflectance of a dark object (i.e. reflectance ~1%).

Chavez (1988) proposed a method which uses the dark object reflectance in one band to predict the corresponding path radiances in all other band_wls. This is done using a relative radiance model which depends on the wavelength and overall atmospheric optical thickness (which is estimated based on the dark object's DN value). This has the advantage that the path radiance is actually correlated across different sensor band_wls and not computed individually for each band using independent dark objects. He proposed a relative radiance model which follows a wavelength dependent scattering that ranges from a power of -4 over -2, -1, -0.7 to -0.5 for very clear over clear, moderate, hazy to very hazy conditions. The relative factors are computed individually for each 1/1000 wavelength within each band range and subsequently averaged over the band as proposed by Goslee (2011).

The atmospheric transmittance towards the sensor (Tv) is approximated by 1.0 (DOS2, Chavez 1996) or Rayleigh scattering (DOS4, Moran et al. 1992)

The atmospheric transmittance from the sun (Tz) is approximated by the cosine of the sun zenith angle (DOS2, Chavez 1996) or again using Rayleigh scattering (DOS4, Moran et al. 1992).

The downwelling diffuse irradiance is approximated by 0.0 (DOS2, Chavez 1996) or the hemispherical integral of the path radiance (DOS4, Moran et al. 1992).

Equations for the path radiance are taken from Song et al. (2001).

For some sensors, the band wavelengths are already included. See lutInfo()[grepl("_BANDS", names(lutInfo()$META))] if your sensor is included. To retrieve a sensor, use lutInfo()$<Sensor ID>_BANDS.

Value

Satellite object with path radiance for each band in the metadata (W m-2 micrometer-1)

Vector object with path radiance values for each band (W m-2 micrometer-1)

References

Chavez Jr PS (1988) An improved dark-object subtraction technique for atmospheric scattering correction of multispectral data. Remote Sensing of Environment 24/3, doi: 10.1016/00344257(88)90019-3.

Chavez Jr PS (1996) Image-based atmospheric corrections revisited and improved. Photogrammetric Engineering and Remote Sensing 62/9, available online at https://www.researchgate.net/publication/236769129_Image-Based_Atmospheric_Corrections_-_Revisited_and_Improved.

Goslee SC (2011) Analyzing Remote Sensing Data in R: The landsat Package. Journal of Statistical Software, 43/4, 1-25, doi: 10.18637/jss.v043.i04.

Moran MS, Jackson RD, Slater PN, Teillet PM (1992) Evlauation of simplified procedures for rretrieval of land surface reflectane factors from satellite sensor output.Remote Sensing of Environment 41/2-3, 169-184, doi: 10.1016/00344257(92)90076V.

Song C, Woodcock CE, Seto KC, Lenney MP, Macomber SA (2001) Classification and Change Detection Using Landsat TM Data: When and How to Correct Atmospheric Effects? Remote Sensing of Environment 75/2, doi: 10.1016/S00344257(00)001693.

If you refer to Sawyer and Stephen 2014, please note that eq. 5 is wrong.

See Also

This function is used by `calcAtmosCorr` to compute the path radiance.

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)
sat <- calcTOAIrradModel(sat)

bds <- "B002n"
val <- calcPathRadDOS(x = min(getValues(getSatDataLayer(sat, bds))),
                      bnbr = getSatLNBR(sat, bds),
                     band_wls = data.frame(LMIN = getSatLMIN(sat, getSatBCDESolar(sat)),
                                           LMAX = getSatLMAX(sat, getSatBCDESolar(sat))),
                      radm = getSatRADM(sat, getSatBCDESolar(sat)),
                      rada = getSatRADA(sat, getSatBCDESolar(sat)),
                      szen = getSatSZEN(sat, getSatBCDESolar(sat)),
                      esun = getSatESUN(sat, getSatBCDESolar(sat)),
                      model = "DOS2",
                      scat_coef = -4)
val
```

`calcTOAIrradModel` *Compute top of atmosphere solar irradiance for sensor bands using LUTs*

Description

Compute mean extraterrestrial solar irradiance (ESun) using tabulated mean solar spectral data and the band specific relative spectral response (rsr) functions.

Usage

```
## S4 method for signature 'Satellite'
calcTOAIrradModel(x, model = "MNewKur", normalize = TRUE, esd)

## S4 method for signature 'data.frame'
calcTOAIrradModel(x, model = "MNewKur", normalize = TRUE, esd)
```

Arguments

x	A Satellite object or the relative spectral response function for the respective band as data.frame (see details for structure).
model	Tabulated solar radiation model to be used (one of MCebKur_MChKur, MNewKur, MthKur, MoldKur, MODWherli_WMO, NN, see reference on tabulated solar irradiance below).
normalize	Logical; if TRUE, ESun is normalized to mean earth-sun distance.
esd	Earth-sun distance (AU, can be estimated using calcEarthSunDist). If x is a Satellite object and esd is not supplied and necessary for normalization, it is tried to take it from the metadata, otherwise it is estimated by the day of the year using calcEarthSunDist.

Details

Computation of ESun is taken from Updike and Comp (2011).

Tabulated values for mean earth-sun distance are taken from the data sources mentioned in the references.

If results should not be normalized to a mean earth-sun distance, the actual earth-sun distance is approximated by the day of the year using calcEarthSunDist.

Relative spectral response values have to be supplied as a data.frame which has at least the following three columns: (i) a column "Band" for the sensor band number (i.e. 1, 2, etc.), (ii) a column "WAVELENGTH" for the WAVELENGTH data in full nm steps, and (iii) a column "RSR" for the response information [0...1].

Value

If x is a Satellite object, a Satellite object with ESun information added to the metadata; if x is a data.frame, a vector containing ESun for the respective band(s).

References

Updike T, Comp C (2011) Radiometric use of WorldView-2 imagery. Technical Note, available online at http://www.pancroma.com/downloads/Radiometric_Use_of_WorldView-2_Imagery.pdf.

Tabulated relative spectral response functions (nm-1) are taken from the spectral viewer of the USGS Landsat FAQ.

Tabulated solar irradiance (W m-2 nm-1) is taken from the National Renewable Energy Laboratory.

See Also

calcTOAIrradTable for tabulated solar irradiance values from the literature or calcTOAIrradRadRef for the computation of the solar irradiance based on maximum radiation and reflection values of the dataset.

See calcEarthSunDist for calculating the earth-sun distance based on the day of the year which is called by this function if ESun should be corrected for actual earth-sun distance.

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)
sat <- calcTOAIrradModel(sat)
getSatESUN(sat)

lut <- lutInfo()
calcTOAIrradModel(lut$L8_RSR, model = "MNewKur", normalize = FALSE,
  esd = calcEarthSunDist("2015-01-01"))
```

---

calcTOAIrradRadRef *Compute top of atmosphere solar irradiance using radiation vs. reflection*

Description

Compute extraterrestrial solar irradiance (ESun) using the actual maximum radiation and reflection values within each band.

Usage

```
## S4 method for signature 'Satellite'
calcTOAIrradRadRef(x, normalize = TRUE, esd)

## S4 method for signature 'numeric'
calcTOAIrradRadRef(x, ref_max, normalize = TRUE, esd)
```

Arguments

x	A Satellite object or the maximum radiance of satellite band(s) as numeric object.
normalize	Logical; if TRUE, ESun is normalized to mean earth-sun distance.
esd	Earth-sun distance (AU, can be estimated using calcEarthSunDist). If x is a Satellite object and esd is not supplied and necessary for normalization, it is tried to take it from the metadata, otherwise it is estimated by the day of the year using calcEarthSunDist.
ref_max	Maximum reflextance of satellite band(s).

Details

The actual solar irradiance is computed using the following formula taken from the GRASS GIS i.landsat.toar module

$$ESun = (pid^2)RADIANCE_MAXIMUM/REFLECTANCE_MAXIMUM$$

where d is the earth-sun distance (in AU) and RADIANCE_MAXIMUM and REFLECTANCE_MAXIMUM are the maximum radiance and reflection values of the respective band. All these parameters are taken from the scene's metadata file if a Satellite object is passed to the function.

By default, the resulting actual ESun will be normalized to a mean earth-sun distance to be compatible with other default results from `calcTOAIrradTable` or `calcTOAIrradModel`.

Value

If x is a Satellite object, a Satellite object with ESun information added to the metadata; if x is numeric, a vector containing ESun for the respective band(s).

See Also

`calcTOAIrradTable` for tabulated solar irradiance values from the literature or `calcTOAIrradModel` for the computation of the solar irradiance based on look-up tables for the sensor's relative spectral response and solar irradiation spectral data.

See `calcEarthSunDist` for calculating the earth-sun distance based on the day of the year which is called by this function if ESun should be corrected for actual earth-sun distance.

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)
sat <- calcTOAIrradModel(sat)
getSatESUN(sat)

calcTOAIrradRadRef(x = getSatRadMax(sat, getSatBCDESolar(sat)),
                   ref_max = getSatRefMax(sat, getSatBCDESolar(sat)),
                   normalize = FALSE,
                   esd = calcEarthSunDist("2015-01-01"))
```

`calcTOAIrradTable` *Get top of atmosphere solar irradiance using readily tabulated values*

Description

Get mean extraterrestrial solar irradiance (ESun) using published values.

Usage

```
## S4 method for signature 'Satellite'
calcTOAIrradTable(x, normalize = TRUE, esd)

## S4 method for signature 'factor'
calcTOAIrradTable(x, normalize = TRUE, esd)

## S4 method for signature 'character'
calcTOAIrradTable(x, normalize = TRUE, esd)
```

Arguments

x	A Satellite object or sensor id ("LT4, LT5, LE7") as character.
normalize	Logical; if TRUE, ESun is normalized to mean earth-sun distance.
esd	Earth-sun distance (AU, can be estimated using calcEarthSunDist). If x is a Satellite object and esd is not supplied and necessary for normalization, it is tried to take it from the metadata, otherwise it is estimated by the day of the year using calcEarthSunDist.

Details

Currently implemented sensors are Landsat 4, 5 and 7.

If results should not be normalized to a mean earth-sun distance, the actual earth-sun distance is approximated by the day of the year using calcEarthSunDist.

Please note that ESun values are not required for converting Landsat 8 data to reflectance as the corresponding metadata files provide coefficients necessary to convert digital numbers to radiance and reflectance (taken from https://www.gisagmaps.com/landsat-8-atco/.

Value

Satellite object with ESun information added to the metadata

Vector object containing ESun for the respective band(s)

Vector object containing ESun for the respective band(s)

References

Tabulated values of the solar irradiance for all Landsat sensors are taken from https://www.usgs.gov/core-science-systems/nli/landsat/using-usgs-landsat-level-1-data-product.

See Also

calcTOAIrradRadRef for the computation of the solar irradiance based on maximum radiation and reflection values of the dataset or calcTOAIrradModel for the computation of the solar irradiance based on look-up tables for the sensor's relative spectral response and solar irradiation spectral data.

See calcEarthSunDist for calculating the earth-sun distance based on the day of the year which is called by this function if ESun should be corrected for actual earth-sun distance.

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LE07*.TIF"), full.names = TRUE)
sat <- satellite(files)
calcTOAIrradTable(sat)

calcTOAIrradTable(x = "LE7", normalize = FALSE,
                  calcEarthSunDist("2015-01-01"))
```

`calcTopoCorr` *Correct for topographic effects.*

Description

Correct for topographic effects.

Usage

```
## S4 method for signature 'Satellite'
calcTopoCorr(x, mask = TRUE)

## S4 method for signature 'RasterStackBrick'
calcTopoCorr(x, hillsh, cloudmask = NULL, ...)

## S4 method for signature 'RasterLayer'
calcTopoCorr(x, hillsh, cloudmask = NULL, ...)
```

Arguments

`x`	`Satellite` or `Raster*` object.
`mask`	`logical`. If `TRUE`, the cloudmask from the `Satellite` object (if available) will be considered in the regression model.
`hillsh`	A `RasterLayer` created with `hillShade`.
`cloudmask`	A `RasterLayer` in which clouds are masked with NA values, passed to `mask`.
`...`	Additional arguments passed to `writeRaster`.

Details

The method of Civco (1989) is applied on atmospherically corrected bands (if not already available in the Satellite object, `calcAtmosCorr` is performed with its default settings.): First, an analytical hillshade image is created based on a DEM and sun elevation and sun zenith information from the metadata. A regression between the hillshade (independent variable) and each channel is then calculated with consideration of a cloudmask (if available). The regression coefficents are used to calibrate the hillshade raster (for each channel individually). Finally, the calibrated hillshade image is subtracted from the corresponding channel and the mean value of the channel is added.

Value

If x is a Satellite object, a Satellite object with added, topographic corrected layers; if x is a raster::Raster* object, a raster::Raster* object with converted layer(s).

References

CIVCO, D.L. (1989): Topographic normalization of Landsat Thematic Mapper digitalimagery. *Photogrammetric Engineering & Remote Sensing*, 55, 1303-1309.

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)

## dem

files_dem <- list.files(path, pattern = "DEM", full.names = TRUE)
DEM <- raster(files_dem)

sat <- addSatDataLayer(sat, data = DEM, info = NULL, bcde = "DEM", in_bcde="DEM")

## Not run:
sat <- calcTopoCorr(sat)

## End(Not run)
```

---

compFilePathLandsat *Get filename, bands and metadata file for Landsat 7 and 8 standard 1B/T format*

Description

The function compiles the sensor, band, filename and metadata filename information for standard level 1B/T Landsat files.

Usage

```
compFilePathLandsat(files)

sortFilesLandsat(files, id = FALSE)
```

Arguments

files	Path and filename(s) of one or more Landsat band files or, alternatively, one or more Landsat metadata files.
id	logical, defaults to FALSE. Determines whether to return sorted band files (ie default) or sorting order.

Value

`data.frame` containing filepaths, band numbers and metadata filepaths.

If `id = FALSE` (default), sorted band files as `character`, else the corresponding sorting order as `integer`.

Functions

- `sortFilesLandsat`: Sort Landsat band files in ascending order.

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)

compFilePathLandsat(files)

sortFilesLandsat(files)
sortFilesLandsat(files, id = TRUE) # indices
```

---

compMetaLandsat *Get calibration information from Landsat 8 standard level 1B/T filename*

Description

The function scans a Lansat metadata file for various calibration and orbit coefficients as well as some sensor specific data.

Usage

```
compMetaLandsat(files)
```

Arguments

`files`	Path and filename of the Landsat metadata file.

Value

`data.frame` containing the following information for each band/layer:

- DATE date (e.g. 2013-07-07)
- SID sensor id (e.g. LC8)
- SENSOR sensor name (e.g. Landsat 8)
- SGRP sensor group (e.g. Landsat)
- BID band id (e.g. 7)
- BCDE band code (5 digit standard name, e.g B001n)

- SRES spatial resolution of the sensor band (e.g. 30 for 30 m x 30m)
- TYPE type of the sensor band regarding wavelength (e.g. VIS)
- SPECTRUM spectral range regarding radiation source (e.g. solar)
- CALIB type of applied calibration (e.g. SC for scaled counts)
- RID region id (e.g. R00001) for multi region Satellite objects
- RADA addtition coefficient for radiance conversion
- RADM multiplication coefficient for radiance conversion
- REFA addtition coefficient for reflectance conversion
- REFM multiplication coefficient for reflectance conversion
- BTK1 brightness temperature correction parameter
- BTK2 brightness temperature correction parameter
- SZEN sun zenith angle
- SAZM sun azimuth angle
- SELV sun elevation angle
- ESD earth-sun distance (AU)
- LMIN Minimum wavelength of the band (micrometer)
- LMAX Maximum wavelength of the band (micrometer)
- RADMIN Minimum radiance recorded by the band
- RADMAX Maximum radiance recorded by the band
- REFMIN Minimum reflectance recorded by the band
- REFMAX Maximum reflectance recorded by the band
- LNBR Layer number from 1 to n layers
- LAYER Layer name
- FILE Filepath of the data file
- METAFILE Filepath of the metadata file

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
compMetaLandsat(files)
```

convRad2BT *Convert a band's scaled counts to brightness temperature*

Description

Convert a band's radiance values to brightness temperature without any kind of atmospheric correction etc.

Usage

```
## S4 method for signature 'Satellite'
convRad2BT(x)

## S4 method for signature 'RasterStack'
convRad2BT(x, k1, k2)

## S4 method for signature 'RasterLayer'
convRad2BT(x, k1, k2)
```

Arguments

x	An object of class Satellite, raster::RasterStack or raster::RasterLayer providing radiance values.
k1, k2	Temperature correction parameters.

Details

The conversion functions are taken from USGS' Landsat 8 Data Users Handbook which is available online at https://www.usgs.gov/core-science-systems/nli/landsat/landsat-8-data-users-handbook.

Value

If x is a Satellite object, a Satellite object with added converted layers;
if x is a raster::Raster* object, a raster::Raster* object with converted layer(s).

See Also

calcAtmosCorr for converions of scaled counts to physical units including a scene-based atmospheric correction.

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)
sat <- convRad2BT(sat)
```

convRad2Ref *Convert a band's scaled counts or radiance values to reflectance*

Description

Convert a band's scaled counts to reflectance using a simple linear conversion without any kind of atmospheric correction etc.

Usage

```
## S4 method for signature 'Satellite'
convRad2Ref(x, szen_correction = "TRUE")

## S4 method for signature 'RasterStack'
convRad2Ref(x, mult, add, szen)

## S4 method for signature 'RasterLayer'
convRad2Ref(x, mult, add, szen)
```

Arguments

x	An object of class Satellite, raster::RasterStack or raster::RasterLayer providing radiance values.
szen_correction	Logical; if TRUE, sun zenith correction is being applied.
mult	Multiplicative coefficient for value transformation (i.e. slope).
add	Additive coefficient for value transformation (i.e. offset)
szen	Cosine of solar zenith angle.

Details

The conversion functions are taken from USGS' Landsat 8 Data Users Handbook which is available online at https://www.usgs.gov/core-science-systems/nli/landsat/landsat-8-data-users-handbook.

If the sensor does not provide linear conversion coefficients for reflectance computation, the reflectance is calculated using the solar irradiance following the functions taken from USGS' Landsat 7 manual, chapter 11.3.2, which is available online at https://www.usgs.gov/media/files/landsat-7-data-users-handbook.

Value

If x is a Satellite object, a Satellite object with added converted layers;
if x is a raster::Raster* object, a raster::Raster* object with converted layer(s).

See Also

calcAtmosCorr for conversions of scaled counts to physical units including a scene-based atmospheric correction.

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)
sat <- convRad2Ref(sat)

# If you use a raster layer, supply required meta information
bcde <- "B002n"
convRad2Ref(x = getSatDataLayer(sat, bcde),
            mult = getSatRADM(sat, bcde),
            add = getSatRADA(sat, bcde))
```

---

convRef2RadLinear *Convert reflectance to radiance using linear function coefficients*

Description

The function converts the reflectance (ref) back to radiance (rad) given that linear conversion coefficients for both radiance and reflectance are available.

Usage

```
convRef2RadLinear(band, refm, refa, radm, rada, szen)
```

Arguments

band	raster::RasterStack or raster::RasterLayer containing reflectance.
refm	Multiplication coefficient for reflectance conversion.
refa	Addtition coefficient for reflectance conversion.
radm	Multiplication coefficient for radiance conversion.
rada	Addition coefficient for radiance conversion.
szen	Sun zenith angle.

Details

The conversion functions are taken from USGS' Landsat 8 Data Users Handbook which is available online at https://www.usgs.gov/core-science-systems/nli/landsat/landsat-8-data-users-handbook.

Value

raster::Raster* object with converted values.

convSC2Rad *Convert a band's scaled counts to radiance*

Description

Convert a band's scaled counts to radiance using a simple linear conversion without any kind of atmospheric correction etc.

Usage

```
## S4 method for signature 'Satellite'
convSC2Rad(x, szen_correction = "TRUE", subset = FALSE)

## S4 method for signature 'RasterStack'
convSC2Rad(x, mult, add, szen)

## S4 method for signature 'RasterLayer'
convSC2Rad(x, mult, add, szen)
```

Arguments

x	An object of class Satellite, raster::RasterStack or raster::RasterLayer providing scaled counts (DNs).
szen_correction	Logical; if TRUE, sun zenith correction is being applied.
subset	Logical; if TRUE, all layers but the cropped ones are being dropped; if FALSE (default), cropped layers are appended to the Satellite object.
mult	Multiplicative coefficient for value transformation (i.e. slope).
add	Additive coefficient for value transformation (i.e. offset).
szen	Cosine of solar zenith angle.

Details

The conversion functions are taken from USGS' Landsat 8 Data Users Handbook which is available online at https://www.usgs.gov/core-science-systems/nli/landsat/landsat-8-data-users-handbook.

Value

If x is a Satellite object, a Satellite object with added converted layers;
if x is a raster::Raster* object, a raster::Raster* object with converted layer(s).

See Also

calcAtmosCorr for conversions of scaled counts to physical units including a scene-based atmospheric correction.

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)
sat <- convSC2Rad(sat)

# If you use a raster layer, supply required meta information
bcde <- "B002n"
convSC2Rad(x = getSatDataLayer(sat, bcde),
           mult = getSatRADM(sat, bcde),
           add = getSatRADA(sat, bcde))
```

---

convSC2Ref *Convert a band's scaled counts or radiance values to reflectance*

Description

Convert a band's scaled counts to reflectance using a simple linear conversion without any kind of atmospheric correction etc.

Usage

```
## S4 method for signature 'Satellite'
convSC2Ref(x, szen_correction = "TRUE", subset = FALSE)

## S4 method for signature 'RasterStack'
convSC2Ref(x, mult, add, szen)

## S4 method for signature 'RasterLayer'
convSC2Ref(x, mult, add, szen)
```

Arguments

x	An object of class Satellite, raster::RasterStack or raster::RasterLayer providing scaled counts (DNs).
szen_correction	Logical; if TRUE, sun zenith correction is being applied.
subset	Logical; if TRUE, all layers but the cropped ones are being dropped; if FALSE (default), cropped layers are appended to the Satellite object.
mult	Multiplicative coefficient for value transformation (i.e. slope).
add	Additive coefficient for value transformation (i.e. offset).
szen	Cosine of solar zenith angle.

Details

The conversion functions are taken from USGS' Landsat 8 Data Users Handbook which is available online at https://www.usgs.gov/core-science-systems/nli/landsat/landsat-8-data-users-handbook.

If the sensor does not provide linear conversion coefficients for reflectance computation, the reflectance is calculated using the solar irradiance following the functions taken from USGS' Landsat 7 manual, chapter 11.3.2, which is available online at https://www.usgs.gov/media/files/landsat-7-data-users-handbook.

Value

If x is a Satellite object, a Satellite object with added converted layers;
if x is a `raster::Raster*` object, a `raster::Raster*` object with converted layer(s).

See Also

`calcAtmosCorr` for conversions of scaled counts to physical units including a scene-based atmospheric correction.

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)
sat <- convSC2Ref(sat)

# If you use a raster layer, supply required meta information
bcde <- "B002n"
convSC2Ref(x = getSatDataLayer(sat, bcde),
           mult = getSatRADM(sat, bcde),
           add = getSatRADA(sat, bcde))
```

---

crop *Crop Satellite object*

Description

The function is a wrapper around the `crop` function to easily crop a Satellite object by an `extent` object.

Usage

```
## S4 method for signature 'Satellite'
crop(x, y, subset = TRUE, snap = "near")
```

Arguments

x	Satellite object.
y	extent object.
subset	Logical; if TRUE (default), all layers but the cropped ones are being dropped; if FALSE, cropped layers are appended to the Satellite object.
snap	Direction towards which to align the extent as character. Available options are "near" (default), "in" and "out" (see alignExtent).

Details

Crop layers of a Satellite object to the size of a given raster::extent object.

Value

A Satellite object consisting of cropped layers only. If subset = FALSE, a Satellite object with the cropped layers appended.

References

Please refer to the respective functions for references.

See Also

This function is a wrapper for raster::crop.

Examples

```
## Not run:
## sample data
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)

## geographic extent of georg-gassmann-stadium (utm 32-n)
ext_ggs <- raster::extent(484015, 484143, 5627835, 5628020)

## crop satellite object by specified extent
sat_ggs <- crop(sat, ext_ggs)

plot(sat)
plot(sat_ggs)

## End(Not run)
```

demTools *Compute terrain characteristics from digital elevation models*

Description

Compute terrain characteristics from digital elevation models (DEM) using `raster::terrain` or `raster::hillShade`.

Usage

```
## S4 method for signature 'Satellite'
demTools(x, method = "hillShade", bcde = "DEM")

## S4 method for signature 'RasterLayer'
demTools(x, sunElev, sunAzim, method = "hillShade")
```

Arguments

x	A DEM provided as an object of class Satellite or RasterLayer.
method	Currently "slope", "aspect" and "hillshade" are implemented.
bcde	The name of the DEM layer in the Satellite object.
sunElev	If `method = "hillShade"`, the elevation angle of the sun in degrees. See parameter `angle` in `hillShade`.
sunAzim	If `method = "hillShade"`, the sun azimuth angle in degree. See parameter `direction` in `hillShade`.

Value

If x is a Satellite object, a Satellite object with added layer containing calculated terrain information; if x is a `raster::RasterLayer` object, a `raster::RasterLayer` object with calculated terrain information.

See Also

`raster::terrain`, `raster::hillShade`.

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)

## dem
files_dem <- list.files(path, pattern = "DEM", full.names = TRUE)
DEM <- raster(files_dem)

sat <- addSatDataLayer(sat, data = DEM, info = NULL, bcde = "DEM", in_bcde="DEM")
sat <- demTools(sat)
```

extend *Extend a Satellite object*

Description

The function is a wrapper around extend to easily extend a Satellite object to a larger spatial extent.

Usage

```
## S4 method for signature 'Satellite'
extend(x, y, subset = TRUE, value = NA)
```

Arguments

x	Satellite object.
y	Target Extent, see extent.
subset	Logical. If TRUE (default), all layers but the extended ones are being dropped, else the extended layers are appended to the initial Satellite object.
value	Fill value assigned to new cells passed to extend, defaults to NA.

Value

A Satellite object consisting of extended layers only or, if subset = FALSE, a Satellite object with the extended layers appended.

See Also

This function is a wrapper around extend.

Examples

```
## Not run:
## sample data
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)

## geographic extent of georg-gassmann-stadium (utm 32-n)
ext_ggs <- raster::extent(482606.4, 482781.4, 5627239, 5627489)

## extend satellite object by specified extent
sat_ggs <- extend(sat, ext_ggs)

plot(sat)
plot(sat_ggs)

## End(Not run)
```

l7 *Landsat 7 sample data*

Description

This dataset comes from the USGS. It contains part of the Landsat 7 scene LE07_L1TP_195025_20010730_20170204_01_T1 (Collection 1 Level-1) from 2001-07-30 over Maburg, Germany.

Format

`RasterStack` with bands 1-8 (incl. QA) of 41 by 41 pixels.

Details

Use of this data requires your agreement to the USGS regulations on using Landsat data.

Source

https://earthexplorer.usgs.gov/

Examples

```
plotRGB(l7, r = 3, b = 1, stretch = "hist")
```

l8 *Landsat 8 sample data*

Description

This dataset comes from the USGS. It contains part of the Landsat 8 scene LC08_L1TP_195025_20130707_20170503_01_T1 (Collection 1 Level-1) from 2013-07-07 over Maburg, Germany.

Format

`RasterStack` with bands 1-7, 9-11 (incl. QA) of 41 by 41 pixels.

Details

Use of this data requires your agreement to the USGS regulations on using Landsat data.

Source

https://earthexplorer.usgs.gov/

Examples

```
plotRGB(l8, r = 4, g = 3, b = 2, stretch = "hist") # true-color composite
plotRGB(l8, r = 5, g = 4, b = 3, stretch = "hist") # false-color composite
```

lutInfo *Get or access internal LUT values used by various functions*

Description

Get internal look-up table (LUT) values from sysdata.rda which have been compiled using data-raw/lut_data.R. Metadata is stored in `lut$meta`.

Usage

```
lutInfo()

lutInfoBandsFromSID(sid)

lutInfoSensorFromSID(sid)

lutInfoBCDEFromBID(sid, bid)

lutInfoBIDFromBCDE(bcde, sid)

lutInfoRSRromSID(sid)

lutInfoSIDfromFilename(files)

lutInfoSGRPfromFilename(file)
```

Arguments

`sid`	Sensor id as returned e.g. from `lutInfoSensorFromSID`.
`bid`	Band id as returned e.g. from `lutInfoBIDFromBCDE`.
`bcde`	Band code as returned e.g. from `lutInfoBCDEFromBID`.
`files`	Filename (or filepath) of one or more remote sensing data filenames
`file`	Filename of a remote sensing data file

Details

The functions above return the following information:

- `lutInfoBandsFromSID` returns the band info block.
- `lutInfoBCDEFromBID` returns the band code.
- `lutInfoBIDFromBCDE` returns the band ids.
- `lutInfoRSRromSID` returns the relative spectral response (rsr) for the sensor.
- `lutInfoSensorFromSID` returns the sensor name.

The LUT contains the following band information taken, if not specified otherwise, from the USGS Landsat FAQ:

l4_band_wl Minimum/maximum wavelength for Landsat 4 bands.

l5_band_wl Minimum/maximum wavelength for Landsat 5 bands.

l7_band_wl Minimum/maximum wavelength for Landsat 7 bands.

l8_band_wl Minimum/maximum wavelength for Landsat 8 bands.

l7_rsr Landat 7 rsr (nm-1) taken from the spectral viewer of the USGS Landsat FAQ.

l8_rsr Landat 8 rsr (nm-1) taken from the spectral viewer of the USGS Landsat FAQ.

solar Solar irradiance (W m-2 nm-1) taken from the National Renewable Energy Laboratory.

l7_esun Tabulated ESun values from tab 11.3 (Thuillier spectrum) of the Landsat7 handbook.

l5_esun, l4_esun Tabulated ESun values from Chander G, Markham B (2003) Revised Landsat-5 TM radiometric calibration procedures and postcalibration dynamic ranges. IEEE Transaction on Geoscience and Remote Sensing 41/11, doi: 10.1109/LGRS.2007.898285.

Value

List containing several `data.frame` objects with LUT values.

Functions

- `lutInfoBandsFromSID`:
- `lutInfoSensorFromSID`:
- `lutInfoBCDEFromBID`:
- `lutInfoBIDFromBCDE`:
- `lutInfoRSRromSID`:
- `lutInfoSIDfromFilename`:
- `lutInfoSGRPfromFilename`:

Examples

```
ls_li <- lutInfo()
# str(ls_li)
```

`maskInvarFeatures` *Identify pseudo-invariant features from a satellite scene*

Description

Identify pseudo-invariant features from a satellite scene based on a vis, near infravis and short-wave infravis band.

Usage

```
## S4 method for signature 'Satellite'
maskInvarFeatures(x)

## S4 method for signature 'RasterStack'
maskInvarFeatures(x, quant = 0.01, id_vis = 1L, id_nir = 2L, id_swir = 3L)

## S4 method for signature 'RasterLayer'
maskInvarFeatures(x, nir, swir, quant = 0.01)
```

Arguments

x	A Satellite object or a raster::RasterLayer providing the sensor's vis band.
quant	A value v = [0...1] which is used to define the percentage threshold values (thv) for invariant features (nir/vis ratio < thv, swir band values > 1-thv).
id_vis	Index of the visible band.
id_nir	Index of the near infravis band.
id_swir	Index of the short-wave infravis band.
nir	A raster::RasterLayer containing the sensor's nir band.
swir	A raster::RasterLayer containing the sensor's swir band.

Details

Invariant features are identified as pixels which belong to the group of (i) the n lowest VIS/NIR ratios and of (ii) the highest n SWIR values. The value of n is given by the parameter quant = [0...1].

Value

If x is a Satellite object, a Satellite object with added layer;
if x is a raster::RasterLayer object, a a raster::RasterLayer object with added layers (1 indicates invariant pixels, 0 otherwise).

References

This function is taken and only slightly modified from the PIF function by Sarah C. Goslee (2011). Analyzing Remote Sensing Data in R: The landsat Package. Journal of Statistical Software,43(4), 1-25, doi: 10.18637/jss.v043.i04.

The underlying theory has been published by Schott RJ, Salvaggio C and Volchok WJ (1988) Radiometric scene normalization using pseudoinvariant features. Remote Sensing of Environment 26/1, doi: 10.1016/00344257(88)901162.

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)
```

```
sat <- maskInvarFeatures(sat)

maskInvarFeatures(x = getSatDataLayer(sat, "B004n"),
                  nir = getSatDataLayer(sat, "B005n"),
                  swir = getSatDataLayer(sat, "B007n"))

## when dealing with a 'RasterStack'
rst <- stack(files[c(6, 7, 9)])
maskInvarFeatures(rst)
```

`names` *Get/set Satellite data layer names*

Description

Get/set Satellite data layer names, i.e. the BCDE id.

Usage

```
## S4 method for signature 'Satellite'
names(x)

## S4 replacement method for signature 'Satellite'
names(x) <- value
```

Arguments

`x`	A Satellite object.
`value`	Band codes of the individual data layers.

Value

Satellite data layer names as character vector.

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)
names(sat)
new_names <- paste0(names(sat), "_test")
names(sat) <- new_names
```

plot *Plot a Satellite object*

Description

This is the standard plotting routine for the 'Satellite' class. Layers are drawn either from the start (default; limited to a maximum of 16 sub-plots) or according to the speficied band codes.

Usage

```
## S4 method for signature 'Satellite,ANY'
plot(x, bcde = NULL, col = grDevices::grey.colors(100), ...)
```

Arguments

x	A 'Satellite' object, usually returned by satellite.
bcde	Band codes to be visualized, e.g. returned by getSatBCDE. If not supplied, the initial (up to) 16 layers are being visualized.
col	Color scheme.
...	Further arguments passed on to plot.default.

See Also

plot.default, par.

Examples

```
## Not run:
## sample data
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)

## display data without quality flag layer
bds <- getSatBCDE(sat)[1:11]
plot(sat, bcde = bds)

## End(Not run)
```

satellite *Create a Satellite object*

Description

Method to create a Satellite object.

Usage

```
## S4 method for signature 'character'
satellite(x, meta, log)

## S4 method for signature 'Raster'
satellite(x, meta, log)

## S4 method for signature 'list'
satellite(x, meta, log)
```

Arguments

x	A vector of filenames, a (multi-layered) Raster* object or a list of single RasterLayer objects (see raster). In the latter case, be aware that bands must be arranged in ascending order (eg using sortFilesLandsat).
meta	Optional metadata object (e.g. returned from compMetaLandsat). If 'x' is a satellite dataset and recognised as "Landsat", then the metadata is automatically extracted from the respective meta information file if both the satellite data and the metadata file follow the USGS Earth Explorer's naming convention.
log	Optionally supply a log entry.

Details

A Satellite object consists of three data sections: (i) a raster data section which holds the actual data values of the respective sensor bands, (ii) a metadata grid which holds meta information for each sensor band (e.g. calibration coefficients, type of sensor band etc.) and (iii) a list of log information which records the processing history of the entire dataset.

Value

A Satellite object.

See Also

(i) compMetaLandsat to get more information about the structure of the metadata component; (ii) https://www.usgs.gov/faqs/what-naming-convention-landsat-collections-level-1-scenes?qt-news_science_products=0#qt-news_science_products for detailed information about the naming conventions for Landsat scene identifiers; and (iii) sortFilesLandsat for automated rearrangement of Landsat band files.

Examples

```
## 'character' input (i.e. filenames)
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)

satellite(files)

## raster::RasterStack input
satellite(l8)
```

Satellite-class *An S4 class to represent a complete satellite dataset*

Description

An S4 class to represent a complete satellite dataset

SatelliteInfo-class *An S4 class to represent a satellite data file*

Description

An S4 class to represent a satellite data file

Slots

`name` name of the data file without extension

`filepath` full path and file of the data file

`path` path to the data file

`file` filename incl. extension of the data file

`extension` extension of the data file

SatelliteLayers-class *An S4 class to represent satellite data*

Description

An S4 class to represent satellite data

Slots

`layers` a list object containing individual RasterLayer objects

SatelliteLog-class *An S4 class to represent satellite log data*

Description

An S4 class to represent satellite log data

Slots

`log` a list object containing information on individual processing steps

SatelliteMetaData-class *An S4 class to represent satellite metadata*

Description

An S4 class to represent satellite metadata

Slots

`meta` a data frame object containing the data

satInfo *Get or access Satellite object information used by various functions*

Description

Get information from class Satellite.

Usage

```
getSatDataLayers(sat, bcde = NULL)

getSatDataLayer(sat, bcde)

getSatMeta(sat, bcde)

getSatMetaBCDETemplate(sat, bcde)

getSatLog(sat)

setSatBCDE(sat, bcde)
```

```
createSatBCDE(sat, width = 3, flag = 0, prefix = "B", postfix = "n")

addSatMetaParam(sat, meta_param)

addSatMetaEntry(sat, meta_param)

addSatLog(
  sat,
  info = NA_character_,
  in_bcde = NA_character_,
  out_bcde = NA_character_
)

addSatDataLayer(sat, bcde, data, meta_param, info, in_bcde)

addRasterMeta2Sat(sat)

createRasterMetaData(rst)

updateRasterMetaData(sat, bcde)

countSatDataLayers(sat)

getSatParam(sat, param, bcde, return_bcde = TRUE)

getSatBCDE(sat, lnbr)

getSatBID(sat, bcde)

getSatSID(sat)

getSatSensor(sat)

getSatSensorGroup(sat)

getSatSensorInfo(sat)

getSatSpectrum(sat, bcde)

getSatBCDESolar(sat)

getSatBCDEThermal(sat)

getSatXRes(sat, bcde)

getSatYRes(sat, bcde)
```

```
getSatRes(sat, bcde)

getSatType(sat, bcde)

getSatCalib(sat, bcde)

getSatBCDEType(sat, bcde, type)

getSatBCDEFromType(sat, type = "VIS")

getSatBCDEFromSpectrum(sat, spectrum = "solar")

getSatBCDESres(sat, bcde, type)

getSatBCDECalib(sat, bcde, calib)

getSatBCDESolarCalib(sat, bcde, calib)

getSatBCDEThermalCalib(sat, bcde, calib)

getSatBandInfo(sat, bcde, return_calib = TRUE)

getSatRadMax(sat, bcde)

getSatRadMin(sat, bcde)

getSatRefMax(sat, bcde)

getSatRefMin(sat, bcde)

getSatESD(sat)

getSatESUN(sat, bcde)

getSatSZEN(sat, bcde)

getSatSAZM(sat, bcde)

getSatSELV(sat, bcde)

getSatMetaLayer(sat, bcde)

getSatLayerfromData(sat, bcde, nbr)

getSatLNBR(sat, bcde)

getSatLMIN(sat, bcde)
```

```
getSatLMAX(sat, bcde)

getSatRADA(sat, bcde)

getSatRADM(sat, bcde)

getSatREFA(sat, bcde)

getSatREFM(sat, bcde)

getSatBTK1(sat, bcde)

getSatBTK2(sat, bcde)

getSatPRAD(sat, bcde)

getSatDATE(sat, bcde)

getSatProjection(sat, bcde)
```

Arguments

sat	Satellite object (see satellite).
bcde	Band code.
width, flag	Field width and format modifier for automated creation of BCDE information, defaults to '3' and '0', respectively. See formatC for further details.
prefix, postfix	Prefix and postfix to be added to the created BCDE information.
meta_param	Metadata parameters used to document new data layer
info	Log information added to metadata
in_bcde	BCDE of layer used as input dataset
out_bcde	BCDE of layer used as output dataset
data	Data layer of a Satellite object
rst	Input raster::Raster* object from which to extract metadata.
param	Parameter of the metadata set (i.e. colname)
return_bcde	Return bcde as attribute (TRUE/FALSE)
lnbr	Layer number
type	Type of the sensor band
spectrum	Spectral region, e.g. "solar" or "thermal".
calib	Calibration information.
return_calib	Return calibration information (TRUE/FALSE)
nbr	Return specific data layer selected by number

Details

The functions are generally self-explaining in that sence that `get*` returns the respective information and `set*` sets the respective information from/in the Satellite object.

`addSatLog` adds a log entry to the Satellite object.

Value

Objects of respective type (see `satellite`).

Functions

- `getSatDataLayers`: Return Satellite data layers
- `getSatDataLayer`: Return Satellite data layer i
- `getSatMeta`: Return Satellite object metadata
- `getSatMetaBCDETemplate`: Return template for Satellite object metadata which is based on existing band
- `getSatLog`: Return Satellite object log info
- `setSatBCDE`: Set BCDE/data layer names of a Satellite object
- `createSatBCDE`: If not supplied, automatically create BCDE names of a Satellite object
- `addSatMetaParam`: Add additional or overwrite metainformation parameter to Satellite object
- `addSatMetaEntry`: Add metainformation for an additional layer to Satellite object
- `addSatLog`: Add new log entry to Satellite object
- `addSatDataLayer`: Add new Satellite data layer
- `addRasterMeta2Sat`: Add raster meta data to Satellite object meta data
- `createRasterMetaData`: Create raster meta data
- `updateRasterMetaData`: Create raster meta data
- `countSatDataLayers`: Return number of Satellite data layers
- `getSatParam`: Return parameter (general method implemented by the specific functions below)
- `getSatBCDE`: Return Band code
- `getSatBID`: Return Band IDs
- `getSatSID`: Return sensor ID
- `getSatSensor`: Return sensor
- `getSatSensorGroup`: Return sensor group
- `getSatSensorInfo`: Return sensor information
- `getSatSpectrum`: Return spectrum
- `getSatBCDESolar`: Return solar band codes
- `getSatBCDEThermal`: Return thermal band codes
- `getSatXRes`: Return sensor x resolution
- `getSatYRes`: Return sensor y resolution

- getSatRes: Return mean sensor resolution (mean of x and y res)
- getSatType: Return sensor type
- getSatCalib: Return calibration level
- getSatBCDEType: Return TYPE band codes
- getSatBCDEFromType: Return BCDE matching TYPE
- getSatBCDEFromSpectrum: Return BCDE matching TYPE
- getSatBCDESres: Return the mean of x and y resolution for band codes matching type
- getSatBCDECalib: Return calibration level for band codes matching type
- getSatBCDESolarCalib: Return calibration level for band codes machting type and are solar bands
- getSatBCDEThermalCalib: Return calibration level for band codes machting type and are thermal bands
- getSatBandInfo: Return band information
- getSatRadMax: Return maximum radiance for bcde
- getSatRadMin: Return minimum radiance for bcde
- getSatRefMax: Return maximum reflectance for bcde
- getSatRefMin: Return minimum reflectance for bcde
- getSatESD: Return earth-sun distance
- getSatESUN: Return actual solar TOA irradiance
- getSatSZEN: Return sun zenith angle
- getSatSAZM: Return sun azimuth angle
- getSatSELV: Return Sun elevation
- getSatMetaLayer: Return Layer name from metadata
- getSatLayerfromData: Return Layer name from data layer
- getSatLNBR: Return Layer number
- getSatLMIN: Return minimum wavelength of the sensor band
- getSatLMAX: Return maximum wavelength of the sensor band
- getSatRADA: Return addition coefficient for SC to radiance conversion
- getSatRADM: Return multiplicative coefficient for SC to radiance conversion
- getSatREFA: Return addition coefficient for SC to reflectance
- getSatREFM: Return multiplicative coefficient for SC to reflectance
- getSatBTK1: Return calibration coefficent to convert SC to brightness temperature
- getSatBTK2: Return calibration coefficent to convert SC to brightness temperature
- getSatDATE: Return DATE
- getSatProjection: Return projection

Examples

```
# List of input files
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)

# Raster stack l8
sat <- satellite(l8)
```

---

stack *Convert selected layers of a Satellite object to a RasterStack*

Description

Convert selected layers of a Satellite object to a RasterStack

Usage

```
## S4 method for signature 'Satellite'
stack(x, layer = names(x), ...)
```

Arguments

x	an object of class 'Satellite'
layer	character vector (bcde codes) or integer vector (index) of the layers to be stacked
...	additional arguments passed on to stack

Examples

```
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)

stck <- stack(sat, c("B001n", "B002n", "B003n"))
stck
```

subset *Subset of Satellite object data layers*

Description

Create a subset of data layers from a Satellite object and return it as a standalone Satellite object.

Usage

```
## S4 method for signature 'Satellite'
subset(x, sid, cid)

## S4 method for signature 'Satellite,ANY,ANY'
x[[i]]
```

Arguments

x	Satellite object providing the source band(s) to be adjusted.
sid	Band numbers or bcde which should be extracted
cid	Calibration information used for subsetting (only works if sid is not supplied to the function)
i	Layer index(es) for subsetting.

Value

A Satellite object

A Satellite object

A Satellite object

Examples

```
## sample data
path <- system.file("extdata", package = "satellite")
files <- list.files(path, pattern = glob2rx("LC08*.TIF"), full.names = TRUE)
sat <- satellite(files)

sat[[2:5]]
subset(sat, cid = "SC")
```

Index