Package ‘soil.spec’

February 20, 2015

Type Package

Title Soil Spectroscopy Tools and Reference Models

Version 2.1.4

Date 2014-12-17

Author Andrew Sila, Tomislav Hengl and Thomas Terhoeven-Urselmans

Maintainer Andrew Sila <a.sila@cgiar.org>

URL http://worldagroforestry.org/research/land-health,
http://www.africasoils.net/

Description Methods and classes for processing and analyzing soil and plant infrared (MIR, alpha-MIR and VISNIR) spectroscopy readings based on the Africa Soil Information Services (AfSIS) project data.

Depends R (>= 3.0)

Imports methods, stats, pls, KernSmooth, wavelets, hexView, sp, GSIF

Suggests e1071, class, chemometrics, plyr, plotKML, mgcv, nlme, spatstat, scales, date, lava, rgdal

License GPL (>= 2)

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2014-12-17 13:07:11

R topics documented:

`afspec` *Sample soil spectroscopy data from Africa Sentinel Sites*

Description

Sample data (113 samples) containing absorbances in the Mid infrared region (MIR) collected at 57 sites (Africa Sentinel Sites).

Usage

```
data(afspec)
```

Format

The `afspec` data set contains two data frames — `"samples"` and `"ab"`. `"samples"` table contains the following columns:

- `SAMPLEID` factor; unique soil sample ID (duplicate IDs are possible as same soil sample can be measured twice)
- `Cluster` integer; Africa Sentinel Sites cluster number
- `Plot` integer; Africa Sentinel Sites plot number
- `Depth` factor; lower depth of the sampled horizon
- `Site` factor; site name
- `Country` factor; country name
- `Region` factor; region name
- `Longitude` numeric; longitude in decimal degrees on the WGS84 datum (`Lat` in the LDSF Field Database)
- `Latitude` numeric; latitude in decimal degrees on the WGS84 datum (`Lat` in the LDSF Field Database)

`"ab"` table contains the following columns:

- `SAMPLEID` factor; ...
- `m4003.5..m599.8` numeric; mid-infrared raw absorbance for denoted wavenumbers (Bruker-LTS instrument)
- **...** other raw absorbance values

Author(s)

Africa Soil Information Service (http://africasoils.net)

References

- Vagen, T., Shepherd, K. D., Walsh, M. G., Winowiecki, L., Desta, L. T., & Tondoh, J. E. (2010) AfSIS technical specifications: soil health surveillance. World Agroforestry Centre, Nairobi, Kenya.
- World Agroforestry Centre (http://worldagroforestry.org)

Examples

```
library(sp)
library(rgdal)
data(afspec)
sp <- afspec$samples[,c("Longitude","Latitude")]
coordinates(sp) <- ~Longitude+Latitude
proj4string(sp) <- CRS("+proj=longlat +datum=WGS84")
## prepare 'samples' table
samples <- cbind(afspec$samples["SAMPLEID"],
    MID="AfSIS-MIR",
    DateTime=Sys.time())
## convert to "SpectraPoints"
afspec.sp <- SpectraPoints(Spectra=Spectra(samples, afspec$ab), sp=sp)
summary(afspec.sp)
get("attributes", spec.opts)
## predict soil pH:
data(m.PHIHOX)
PHIHOX <- predict(afspec.sp, model=m.PHIHOX)
data(m.ORCCNS)
ORCCNS <- predict(afspec.sp, model=m.ORCCNS)
## Not run:
library(plotKML)
afspec.pnts <- SpatialPointsDataFrame(sp, cbind(PHIHOX, ORCCNS))
data(SAGA_pal)
plotKML(afspec.pnts["PHIHOX"], colour_scale=rev(SAGA_pal[[1]]))
plotKML(afspec.pnts["ORCCNS"], colour_scale=SAGA_pal[[1]])
## plot with global soil legends:
library(GSIF)
data(soil.legends)
brks = c(soil.legends[["PHIHOX"]]$MIN[1]/10, soil.legends[["PHIHOX"]]$MAX/10)
pH.range <- cbind(soil.legends[["PHIHOX"]]$MAX, soil.legends[["PHIHOX"]]$MIN)
levs = paste(signif(rowMeans(pH.range)/10, 2))
afspec.pnts$Col <- cut(afspec.pnts$PHIHOX, breaks=brks, labels=levs)
plotKML(afspec.pnts["Col"], colour_scale=soil.legends[["PHIHOX"]]$COLOR)

## End(Not run)
```

afss	*Africa Sentinel Site data (wet chemistry)*

Description

Africa soil sampled collected at AfSIS sentinel sites. Part of the African Land Degradation Surveillance Framework (LDSF) Field Database.

Usage

```
data(afss)
```

Format

The `afss` data set contains two data frames — sites and horizons. Sites table contains the following columns:

`SOURCEID` factor; study soil profile ID (`SPID` in the LDSF Field Database)

`CNTISOC` factor; ISO country code (`CC` in the LDSF Field Database)

`LOCNAME` factor; sentinel site name referring to the closest populated place (`Site` in the LDSF Field Database)

`LONWGS84` numeric; longitude in decimal degrees on the WGS84 datum (`Lon` in the LDSF Field Database)

`LATWGS84` numeric; latitude in decimal degrees on the WGS84 datum (`Lat` in the LDSF Field Database)

Horizons table contains the following columns:

`SOURCEID` factor; study soil profile ID (`SPID` in the LDSF Field Database)

`SAMPLEID` factor; soil sample ID (`SSID` in the LDSF Field Database)

`UHDICM` numeric; upper sampling depth from the surface in cm

`LHDICM` numeric; lower sampling depth from the surface in cm

`ORCCNS` numeric; soil organic carbon content in permille (`SOC` in the LDSF Field Database)

`PHIHOX` numeric; pH index measured in water solution (`pH` in the LDSF Field Database)

`EXB` numeric; Exchangeable bases in ppm (`ExBas` in the LDSF Field Database)

`ALUM3S` numeric; Aluminium concentration in ppm (`m3.Al` in the LDSF Field Database)

`ECAM3S` numeric; Exchangeable Ca in cmolc/kg (`ExCa` in the LDSF Field Database)

`EXKM3S` numeric; Exchangeable K in cmolc/kg (`ExK` in the LDSF Field Database)

`EMGM3S` numeric; Exchangeable Mg in cmolc/kg (`ExMg` in the LDSF Field Database)

`ENAM3S` numeric; Exchangeable Mg in cmolc/kg (`ExNa` in the LDSF Field Database)

`NITCNS` numeric; Exchangeable N as volumetric percentage (`Total.Nitrogen` in the LDSF Field Database)

`SNDLDF` numeric; weight percentage of the sand particles (0.05–2.0 mm; `Sand` in the LDSF Field Database)

`afss.spec` table contains the following columns:

`SAMPLEID` factor; soil sample ID (`SSID` in the LDSF Field Database)

`m4003.5` numeric; mid-infrared raw absorbance for denoted wavenumbers (Bruker-LTS instrument)

... other raw absorbance values

Note

The soil samples from the sentinel site network were collected using three stage sampling scheme: a number of sentinel sites (about 10 by 10 kilometers in size) have been first selected over the whole continent (60 sites in 2012). Each sentinel site is stratied into 16 grid cells (each 1 square kilometer), and sampling cluster centroids are randomly located within the grid cells. Around each centroid, 10 sampling plots are randomly located and soil samples taken (or 160 points per sentinel site). At each sampling plot, soil samples were taken at two depths: 0–20, and 20–50 cm. The block size of each sampling plot is 1000 square meters or 0.1 ha. All of the initial Mehlich-3 nutrient concentrations were in mg/kg (or ppm) units, and were converted to molar equivalent units in cmolc/kg (xCa = Ca/200, xK = K/391, xMg = Mg/121, xNa = Na/230).

Author(s)

Africa Soil Information Service (`http://africasoils.net`)

References

- Vagen, T., Shepherd, K. D., Walsh, M. G., Winowiecki, L., Desta, L. T., & Tondoh, J. E. (2010) AfSIS technical specifications: soil health surveillance. World Agroforestry Centre, Nairobi, Kenya.

Examples

```
library(sp)
library(rgdal)

data(afss)
xy <- afss$sites
coordinates(xy) <- ~ LONWGS84 + LATWGS84
proj4string(xy) <- "+proj=longlat +datum=WGS84"
xy$PHIHOX_d1 <- merge(xy@data, afss$horizons[afss$horizons$UHDICM==0,], all.x=TRUE)$PHIHOX

## Not run:
## plot points in Google Earth:
library(plotKML)
data(R_pal)
plotKML(xy["PHIHOX_d1"], colour_scale=R_pal[["pH_pal"]])

## obtain country borders:
con.admin <- url("http://gsif.isric.org/lib/exe/fetch.php?media=admin.af.rda")
```

```
load(con.admin)
proj4string(admin.af) <- get("ref_CRS", envir = plotKML.opts)
admin.af <- as(admin.af, "SpatialLines")
## overlay and plot points and maps:
plot(admin.af, col="darkgrey", xlim=xy@bbox[1,], ylim=xy@bbox[2,])
points(xy, pch=21, bg="white", cex=.6, col="black")

## obtain MIR measurements for AfSS samples:
con <- url("http://gsif.isric.org/lib/exe/fetch.php?media=afss.spec.rda")
load(con)
str(afss.spec)

## End(Not run)
```

---

fit.SpectraModel-method

Fits a calibration model for soil spectroscopy

Description

Fits a calibration model that can be used to convert absorbances to soil properties. Implements the "plsr" function available from the pls package.

Usage

```
## S4 method for signature 'formula,SpectraPoints,data.frame'
fit(formulaString, sampled, reference,
   idcol = "SAMPLEID", ab.r = get("MIR", envir=spec.opts),
   CO2.band = get("CO2.band", envir=spec.opts), ncomp, ncomp.max=15,
   repl = 5, segment.type = "interleaved", prefix = "X", ...)
## S4 method for signature 'list,SpectraPoints,data.frame'
fit(formulaString, sampled, reference, ...)
```

Arguments

formulaString	"formula"; formula string
sampled	"SpectraPoints"; absorbances
reference	"data.frame"; reference values estimated in laboratory using at the order of magnitude more precise technique
idcol	"character"; column with sample IDs
ab.r	"numeric"; natural range of wavenumbers of interest
CO2.band	"numeric"; range of wavenumbers for the CO2 band
ncomp	"integer"; number of components (if not specified will be determined using cross-validation)
ncomp.max	"integer"; maximum number of components to be used for model fitting

`repl`	`"integer"`; number of repetitions for `"pls::plsr"`
`segment.type`	`"character"`; segment type for `"pls::plsr"`
`prefix`	`"character"`; default prefix to be added to the band names
`...`	other optional arguments

Details

Returns an object of class `"SpectraModel"`, which contains slots: (1) `"variable"` (variable name according to the global soil data registry), (2) `"Space"` (3D space defined by the 1–3 principal components for absorbances), and (3) `"model"` (fitted model; usually of type `pls::plsr` and as an output from the partial least squares regression). Fitting models for large data sets can be time and memory consuming.

Author(s)

Andrew Sila and Tomislav Hengl

References

- Mevik, B.-H., Wehrens, R. (2007) The pls Package: Principal Component and Partial Least Squares Regression in R. Journal of Statistical Software 18(2), 1-24.
- Africa Soil Information Service (`http://africasoils.net`)

See Also

`predict.SpectraPoints`, `spec.opts`

Examples

```
## Not run:
## build a callibration model using AfSIS data:
library(plyr)
library(sp)
library(rgdal)

data(afss)
afss.tbl <- join(afss$sites, afss$horizons)
con <- url("http://gsif.isric.org/lib/exe/fetch.php?media=afss.spec.rda")
load(con)
afss.spec.ab <- Spectra(samples=data.frame(SAMPLEID=afss.spec$SAMPLEID,
   MID="ICR_SOIL", DateTime=Sys.time()), ab=afss.spec)
sel.r <- !afss.tbl$LONWGS84==100
sp <- data.frame(afss.tbl[sel.r,c("LONWGS84","LATWGS84")])
coordinates(sp) <- ~LONWGS84+LATWGS84
proj4string(sp) <- CRS("+proj=longlat")
attr(sp@coords, "dimnames")[[1]] <- afss.tbl$SAMPLEID[sel.r]
## extend to SprectraPoints:
afss.spec.sp <- SpectraPoints(Spectra=afss.spec.ab, sp=sp)
## select bands of interest:
cutspec <- c(600,2350.8,2379.8,4000)
```

```
col.no <- sapply(names(afss.spec.sp@data@ab)[-1],
   function(x){as.numeric(strsplit(x, "X")[[1]][2])})
sel <- col.no>cutspec[1] & col.no<cutspec[2] |
       col.no>cutspec[3] & col.no<cutspec[4]
summary(sel)
formulaString <- as.formula(paste("na.omit(log(ORCDRC)) ~ ",
   paste(names(afss.spec.sp@data@ab[-1])[sel], collapse="+")))
m.ORC <- fit(formulaString, sampled=afss.spec.sp, reference=afss.tbl)
str(m.ORC)

## End(Not run)
## this model, in fact, is included in the package:
data(m.ORCCNS)
str(m.ORCCNS)
```

---

ken.sto — *Sample selection based on the Kennard-Stone algorithm*

Description

The function chooses points based on Euclidean distance measure most representative samples. One can (i) select a number or a percentage of a sample set or (ii) divide a sample set into calibration and representative validation set.

Usage

```
ken.sto(inp, per = "T", per.n = 0.3, num, va = "F", sav = "T", path = "", out = "Sel")
```

Arguments

`inp`	a numerical matrix or data.frame containing the input spectra
`per`	a logical value indicating whether the selected samples should be a percentage (given in `per.n`) or a set number (given in `num`) of `inp`. The default `"T"` takes a percentage.
`per.n`	a numerical value between 0 and 1.
`num`	a numerical value between 1 and the sample number minus 1.
`va`	a logical value indicating whether to select samples out of `inp` or to divide them into a calibration and validation set.
`sav`	a logical value indicating whether the function output shall be saved.
`path`	a character giving the path name where the function output shall be saved.
`out`	a character giving the function output name, in case `sav` is `"T"`.

Details

Sample selection is done following and adapted procedure from Kennard & Stone (1969). It is a stepwise procedure by maximizing the Euclidean distance based on the important number of principal components to the objects already chosen. The number of important principal components is selected so that the increase in cumulative explained variance within the next three components is lower than 4 percent. The starting samples are the two extreme samples (most negative and positive ones) of the important principal components.

`per.n` having a value of 0.4 while `va` equal to `"F"` chooses 40 percent of the sample set. When `va` is equal to `"T"` the validation set comprises 40 percent of the sample set.

A graph is given back showing the selected samples in the principal component space (only the important PC's). This is the same graphic generated by `plot.ken.sto`.

Value

`ken.sto` returns a list with class `"ken.sto"` containing the following components:

`Calibration and validation set`	the logical object `va`.
`Number important PC`	integer giving the number of chosen important components - important for choosing the starting samples.
`PC space important PC`	score value matrix of important principal components.
`Chosen samples names`	chosen sample names when `va` equal to `"F"`
`Chosen row number`	chosen row numbers when `va` equal to `"F"`
`Chosen calibration sample names`	chosen calibration sample names when `va` equal to `"T"`
`Chosen calibration row number`	chosen calibration row numbers when `va` equal to `"T"`
`Chosen validation sample names`	chosen validation sample names when `va` equal to `"T"`
`Chosen validation row number`	chosen validation row numbers when `va` equal to `"T"`

Author(s)

Thomas Terhoeven-Urselmans

References

Kennard, R. W. and Stone, L. A. (1969) *Computer aided design of experiments.* Technometrics 11(1), 137-148.

Examples

```
## Not run: ken.sto(inp, per = "T", per.n = 0.3, num, va = "F", sav = "T", path = "", out = "Sel")
## Not run: plot(ken.sto)(x,...)
```

plot.Spectra *Plots spectral signatures*

Description

Plots spectral signatures. Each line represents a single sample.

Usage

```
## S4 method for signature 'Spectra,ANY'
plot(x, y, ...)
```

Arguments

x	object of class "Spectra"
y	ignored
...	other optional arguments

Details

Wavenumbers are shown on the x-axis at the bottom. The second top x-axis displays the equivalent of wavenumbers interms of wavelengths.

Author(s)

Andrew Sila and Tomislav Hengl

Examples

```
## Not run: ## Original binary Opus files:
pth = system.file(package = "soil.spec")
lst <- as.list(list.files(path=pth, pattern="*.0$", full.names=TRUE))
xx <- read.opus(lst[[1]])
plot(xx@data)

## End(Not run)
```

predict.SpectraPoints-method

Predict soil properties from an object of class "SpectraPoints"

Description

Predicts soil properties from an object of class "SpectraPoints" (or a "data.frame" containing absorbances and sample IDs) either by using a referent "SpectraModel" (comes with the package) or by using the user created model. Extends, for example, the "predict.mvr" function available from the pls package.

Usage

```
## S4 method for signature 'SpectraPoints'
predict(object, idcol = "SAMPLEID", model,
   variable = model@variable,
   output, validate = FALSE, model.class = "mvr",
   confidence.band = TRUE, prob. = .90,
   signif.digits = 3,
   st.wavenumbers=wavenumbers,
   instr.range = c("ten-mir", "alp-mir", "mpa-nir")[1],
   ...)
## S4 method for signature 'data.frame'
predict(object, idcol = "SAMPLEID", ...)
```

Arguments

`object`	object of type "SpectraPoints"
`idcol`	"character"; ID column name
`model`	"SpectraModel"; prediction model
`variable`	"character"; variable name (see 'attributes in `spec.env(show.env = TRUE)`)
`output`	"SoilProfileCollection"; optional output object where the predicted soil properties can be written
`validate`	"logical"; runs validation of overlap of points in feature space (see details)
`model.class`	"character"; prediction model class
`confidence.band`	"logical"; specifies whether confidence bands should be derived for each prediction
`prob.`	"numeric"; probability for confidence bands
`signif.digits`	"integer"; significant digits (all output numbers are typically rounded to three significant digits to save space)
`st.wavenumbers`	"data.frame"; standard wavenumbers with band names and upper / lower limits (usually predefined)

`instr.range`	`"character"`; instrument range
`...`	other optional arguments

Details

When predicting from a `data.frame`, column names must contain wavenumbers and first column must contain sample IDs. Validation of overlap in feature space is done using PCs 1-3 derived from absorbance bands. This method basically compares for each new sample shortest distance (in 3D space; derived using `spatstat::nncross.pp3`) to the cloud of referent points and then prints warning if the distance is larger than two times the largest `nndist` (distance) for the referent point cloud.

Author(s)

Tomislav Hengl and Andrew Sila

References

- Mevik, B.-H., Wehrens, R. (2007) The pls Package: Principal Component and Partial Least Squares Regression in R. Journal of Statistical Software 18(2), 1-24.

See Also

`SpectraPoints-class`, `fit.SpectraModel`

Examples

```
data(afspec)
## predict pH using absorbances directly:
data(m.PHIHOX)
x <- predict(afspec$ab, model=m.PHIHOX)
## Not run:
## how good is this model?
## we look at the calibration data used to produce "m.PHIHOX"
pr <- m.PHIHOX@model$fitted.values
library(plyr)
library(scales)
data(afss)
afss.tbl <- join(afss$sites, afss$horizons)
tbl <- merge(data.frame(predicted=pr, SAMPLEID=attr(pr, "names")),
   afss.tbl[,c("SAMPLEID","PHIHOX")])
v.r <- range(tbl$PHIHOX)
r.square <- summary(lm(as.formula(paste("PHIHOX ~ predicted - 1")),
   tbl))$adj.r.squared
plot(x=tbl$predicted, y=tbl$PHIHOX, asp=1, xlim=v.r, ylim=v.r,
   col=alpha("darkgrey", 0.5), xlab="Predicted", ylab="Measured",
   main=paste("R-square:", formatC(r.square, digits=3), sep=""))
abline(a=0, b=1, lw=2, col="black")

## End(Not run)
```

read.opus-method *Reads binary OPUS files containing spectroscopy*

Description

Reads binary OPUS files containing infrared, near infrared and Raman spectroscopy measurements (absorbances) and generates an object of class "SpectraPoints".

Usage

```
## S4 method for signature 'character'
read.opus(file.name, sp = NULL,
   codes = c("ZFF","RES","SNM","DAT","LWN","FXV","LXV","NPT","MXY","MNY","END","TIM"),
    plot.spectra = FALSE, print.progress = FALSE, speclib = "ICRAF",
   signif.digit = get("signif.digit", spec.opts), MID, st.wavenumbers = wavenumbers)
## S4 method for signature 'list'
read.opus(file.name, ...)
```

Arguments

`file.name`	"character"; file name with extension
`sp`	"SpatialPoints"; optional spatial object with coordinates of points with same sample ID's in the coordinates slot row names
`codes`	"character"; default OPUS codes
`plot.spectra`	"logical"; specifies whether to plot spectral curves every time a new sample is loaded
`print.progress`	"logical"; specifies whether to print progress
`signif.digit`	"integer"; number of significant digits
`speclib`	"character"; specifies which spectral library to base data points
`MID`	"character"; metadata ID
`st.wavenumbers`	"data.frame"; standard wavenumbers with band names and upper / lower limits (usually predefined)
`...`	other optional arguments

Details

This function will read OPUS files from Bruker Optics' Alpha, MPA and Tensor-27 spectrometers. A choice for the type of spectral library to be created from the data tables created is done. To match data points to ICRAF's spectral library set `speclib = "ICRAF"`, otherwise to create own spectral library based on OPUS files being converted set `speclib="New"`.

Note

Setting `speclib = "New"` may produce spectra with non-overlapping points brought about by slight drifts on the equipement and combinining with different columns can cause problems. If one chooses to create their own library, stable data points observed over time should be used to align all future scans.

Note

Reading long list of binary files can be time and memory consuming.

Author(s)

Andrew Sila and Tomislav Hengl

References

- Bruker Coorporation Guide for Infrared Spectroscopy

See Also

`SpectraPoints-class`, `predict.SpectraPoints`, `wavenumbers`

Examples

```
## Original binary Opus files:
pth = system.file(package = "soil.spec")
lst <- as.list(list.files(path=pth, pattern="*.0$", full.names=TRUE))
file.info(lst[[1]])
xx <- read.opus(lst)
str(xx)
## predict pH
data(m.PHIHOX)
s.xx <- predict(xx, model = m.PHIHOX, prob. = .75)
s.xx
## Note: duplicate samples get unique name by default

## predict all standard soil properties:
nm <- get("attributes", spec.opts)
nm
pr.lst <- NULL
for(k in 1:length(nm)){
  data(list=paste("m.", nm[k], sep=""))
  try( pr.lst[[k]] <- predict(xx, variable=nm[k],
  model = get(paste("m.", nm[k], sep="")), prob. = .75) )
}
pr <- do.call(cbind, pr.lst)
str(pr)
```

read.spc — *Reads spectral spc-files into R*

Description

`read.spc` reads binary spectral spc-files from a folder into R. The spectra can be made compatible (see details in `make.comp`) either to the first sample wavebands or to the standard wavebands of the ICRAF spectral lab. Information from the scanning method is gathered to check on spectral comparability. The default has been set to ICRAF spectral bands.

Usage

```
read.spc(loa = "", path = "", sav = "F", out = "Sm", save.as = "workspace", wn = "first")
```

Arguments

`loa`	a character giving the path name where the spc-files are stored. If `""` (default), the spc-files are stored in the current working directory.
`path`	a character giving the path name where the function output shall be saved.
`sav`	a logical value indicating whether the function output shall be saved.
`out`	a character giving the function output name, in case `sav` is `"T"`.
`save.as`	a character vector indicating the format of the saved output. `"workspace"` saves the function output named with `out` as workspace. `"csv.file"` saves the function output as csv-file.
`wn`	a character giving the way how the samples should be made compatible. If `"first"`, all spectra are made compatible to the first read sample. if `"ICRAF"`, all spectra are made compatible to the standard wavebands of the ICRAF spectral lab.

Details

Spectra from the near- and mid-infrared range can be read. In case the spc-files saved in `loa` comprise both ranges the function output is given separate for each range.

The function allows to read only spectra in one go, when they have the same material (e.g. soil or plant), were scanned with the same resolution and have the same zero filling. If there are still small differences in the number of wavebands, the spectra are made compatible depending on the argument `wn`. In case spectra with different materials shall be read, the user has to decide the material via graphical interface.

Value

`read.spc` returns a list with class `"read.spc"` containing the following components:

`spectra`	a numerical matrix containing the read spectra.
`additional.information`	a data frame containing some scanning method details.

`raw.spectra`	a data frame merging additional.information and spectra; columns 2 to 4 from the additional.information table are excluded to conform to the existing structure of library in existence

Author(s)

Andrew Sila and Thomas Terhoeven-Urselmans

Examples

```
## Not run: mir<-read.spc(loa="D:/AfSIS/spc files",sav="F",wn="ICRAF")
## Not run: raw<-mir$raw.spectra #To extract the object with both additional.information and spectra
## Not run: raw[1:4,1:4] #This gives a snapshot of the first four rows and the first columns
```

`spec.env` *Soil.spec package specific environmental variables*

Description

Generates internal environment / package specific parameters and settings that can be later on passed to other functions.

Usage

```
spec.env(MIR = c(390, 7500), NIRS = c(3900, 12500),
   NIRP = c(4000,10000), VISNIR1 = c(420, 960),
   VISNIR2 = c(1020, 1770),
   VISNIR3 = c(1830, 2480),
   icraf.htsxt = c(3578, 7497.964, 599.76),
   icraf.alpha = c(2542, 3998.12872, 399.387991),
   icraf.mpa = c(2307, 12493.2, 3598.69),
   CO2.band = c(2350.8,2379.8), signif.digit = 5,
   attributes = c("ORCCNS", "PHIHOX", "ALUM3S",
   "ECAM3S", "EXKM3S", "EMGM3S", "ENAM3S", "EXB",
   "NITCNS", "SNDLDF"),
   mdnames = c("MID", "Instrument_name", "Instrument_URL",
   "Laboratory_name", "Laboratory_contact", "Laboratory_URL",
   "Material_class", "Wavenumber_conversion", "Wavenlength_unit",
   "Location_error"), show.env = FALSE)
```

Arguments

`MIR`	`"numeric"`, mid-infrared part of spectra
`NIRS`	`"numeric"`, near infrared part of spectra
`NIRP`	`"numeric"`, near infrared part of spectra
`VISNIR1`	`"numeric"`, visible near infrared part of spectra
`VISNIR2`	`"numeric"`, visible near infrared part of spectra

`VISNIR3`	`"numeric"`, visible near infrared part of spectra
`icraf.htsxt`	`"numeric"`, ICRAF MIR part of spectra
`icraf.alpha`	`"numeric"`, ICRAF alpha-MIR part of spectra
`icraf.mpa`	`"numeric"`, ICRAF VISNIR part of spectra
`CO2.band`	`"numeric"`, wavenumbers for the CO2 band
`signif.digit`	`"integer"`, default rounding system for absorbances
`attributes`	`"character"`, standard target variables of interest: `"ORCCNS"` (Organic carbon content in soil estimated using the CNS elemental analyzer), `"PHIHOX"` (soil pH measured in water), `"ALUM3S"` (Aluminium concentration estimated using the Mehlich 3 solution), `"ECAM3S"` (exchangeable Calcium determined using using the Mehlich 3 solution), `"EXKM3S"` (exchangeable Potassium determined using using the Mehlich 3 solution), `"EMGM3S"` (exchangeable Magnesium determined using using the Mehlich 3 solution), `"ENAM3S"` (exchangeable Sodium determined using using the Mehlich 3 solution), `"EXB"` (sum of exchangeable bases), `"NITCNS"` (total nitrogen estimated using estimated using the CNS elemental analyzer), `"SNDLDF"` (weight percentage of particles 50-2 mm size determined using the laser diffraction method)
`mdnames`	`"character"`, standard metadata names
`show.env`	`"logical"`, specifies whether the output environment should be printed

Author(s)

Andrew Sila and Tomislav Hengl

Examples

```
# environmental variables:
spec.env()
get("mdnames", envir = spec.opts)
get("attributes", envir = spec.opts)
```

`spectra.outliers` *A tool for screening spectral outliers based on Multidimensional Scaling (MDS) approach*

Description

Reads a data table containing raw spectra which first is transformed using first derivative from Savitzky-Golay algorithm to take off any effects due to measurement differences. Euclidean distances are computed from the first derivative transformed spectra, other types for metric measure can be used but the function uses Euclidean method only. Multidimensional scaling is then used to represent proximities or Euclidean similarity distances in a more organized structure. Simple visualization are produced on request to show how individual spectral point distribution within the Euclidean space. Mean and standard deviation for the similarity distances are calculated for determining the most similar points and the most dissimilar ones. There are two types of situations covered here. When checking for outliers within an highly homogenous spectral set where, outliers will be marked for point lying one standard deviation away from the mean distance. This is the case when screening for outliers from a reference standard sample used to keep checking on instrument stability. The other situation is when checking for spectral outliers from a large spectra collection. In this case, the confidence interval for dertermining outliers is increased upto three standard deviations from the mean to allow for any normal sample to sample differences. Therefore, in the function specifying whether spectra comes from standard or non.standard is recommended for the correct confindence interval calculation.

Usage

```
spectra.outliers(file.name, spectra="non.standard",
    plots=TRUE, smethod="euclidean", k=5)
```

Arguments

`file.name`	`"character"`; file name with extension
`spectra`	`"character"`; specify whether spectra is from "non.standard" for actual sample spectra or is a "standard" for reference samples.
`plots`	`"logical"`; specifies whether to produce outlier diagnostic plots or not. Default is set to TRUE
`smethod`	`"character"`; specifies metric method for computing similarity distance. Euclidean method is the one supported
`k`	`"integer"`; number of components to visualize the MDS distance

Value

`dist`	n by k matrix with similarity distances, n=rows of spectral table; k=the number of components used defualut is set to 5
`all.outliers`	Row numbers with spectra marked as outliers
`spec.out`	Data frame with outlier spectra
`spec.normal`	Data frame with remaining spectra after removing outlier spectra

Note

There are two options for getting list of outliers:

- Take all the points marked as outliers falling outside the confidence bands explained above
- Get into interactive mode to identify outlying points by clicking all the points appearing as outliers. To stop selection press Ctrl+any keyboard button

But whichever the choice, all points outside the confidence band is given in different colour to aid easy identification. User to decide on whether to use the default outliers ir use the interactive mode

Author(s)

Andrew Sila and Tomislav Hengl

References

- Kruskal, J.B. and M. Wish (1978). Multidimensional Scaling. Sage.
- Cox, T.F., Cox, M.A.A. (2001). Multidimensional Scaling. Chapman and Hall.

`SpectraPoints-class` *A class for spectral absorbance measurements*

Description

A class for spectral absorbance measurements typically in the medium infra-red range (MIR), visible near infra-red (VISMIR) and near-infra-red.

Details

The class expects by default absorbance, and not reflectance values. For large data sets we advise using the wide data format.

Slots

`metadata`**:** object of class `"data.frame"`; metadata table containg relevant information such as `"MID"` (metadata ID), `"Instrument_name"`, `"Instrument_URL"`, `"Laboratory_name"`, `"Laboratory_contact"` (contact person and his/her e-mail address), `"Laboratory_URL"`, `"Material_class"` (e.g. soil or vegetation), `"Wavenumber_conversion"`, `"Wavenlength_unit"`, `"Location_error"`

`data`**:** object of class `"Spectra"`; contains absorbance values in wide or long format

`sp`**:** object of class `"SpatialPoints"`; sampling locations

Methods

summary `signature(obj = "SpectraPoints")`: default summary of the object showing number of bands and points

validate `signature(obj = "SpectraPoints")`: check for overlap in feature space in comparison to the calibration data

Author(s)

Tomislav Hengl and Andrew Sila

See Also

`fit.SpectraModel`

Examples

```
## generate the class:
library(sp)
library(rgdal)

data(afspec)
## get spatial coordinates:
sp <- afspec$samples[,c("Longitude","Latitude")]
coordinates(sp) <- ~Longitude+Latitude
proj4string(sp) <- CRS("+proj=longlat +datum=WGS84")
## prepare 'samples' table
samples <- cbind(afspec$samples["SAMPLEID"],
    MID="AfSIS-MIR",
    DateTime=Sys.time())
## convert to "SpectraPoints"
afspec.sp <- SpectraPoints(Spectra=Spectra(samples, afspec$ab), sp=sp)
summary(afspec.sp)
plot(afspec.sp@data)
data(m.PHIHOX)
```

`trans` *Spectral transformation*

Description

`trans` transforms spectra either by calculation of (i) derivatives, (ii) continuum removal or (iii) wavelet transform. The used functions are (i) `locpoly` function in `"KernSmooth"` package, (ii) `chull` and `approx` functions in `"KernSmooth"` package and (iii) `dwt` function in `"wavelets"` package , respectively.

Usage

```
trans(raw, tr = "derivative", order = 1, gap = 21,
   plot.spectrogram = FALSE)

## S3 method for class 'trans'
plot(x,...)
```

Arguments

`raw`	a matrix containing the raw spectra
`tr`	a character naming the transformation method; see details
`order`	an integer between 0 and 3 giving the order of derivative; the value 0 performs smoothing based on the gap
`gap`	an integer between 1 and 30 defining the smoothing interval in wavebands

`plot.spectrogram`	specifies whether to plot the spectrogram by default
`x`	an object of class `"trans"`
`...`	additional arguments

Details

`spec.type` uses for spectral transformation (i) the `locpoly` function in `KernSmooth` package for derivative calculation, (ii) the `chull` and `approx` functions in `"KernSmooth"` package for continuum removal and (iii) the `dwt` function in `wavelets` package for extraction of wavelet coefficients. Experiences showed for wavelet decomposition that the best ratio of prediction performance and sparse spectral representation is reached when all 128 wavelet coefficients from decomposition level three are taken.

Possible options for `tr` are `"derivative"`, `"continuum removed"` or `"wavelet transformed"`.

Value

`trans` returns a list with class `"trans"` containing the following components:

`raw`	a matrix containing the raw spectra.
`trans`	a matrix containing the transformed spectra.
`transformation`	a character naming the transformation method.

Author(s)

Thomas Terhoeven-Urselmans

`wavenumbers` *Standard bands*

Description

Standard band names with upper and lower limits (wavenumbers)

Usage

```
data(wavenumbers)
```

Format

BAND factor; band names

UPPER numeric; upper value

LOWER numeric; lower value

Details

Standard wavenumbers are required by the `read.opus` function. Each `"SpectraModel"` might require different standard bands.

Author(s)

Andrew Sila and Tomislav Hengl

Examples

```
data(m.PHIHOX)
## get bands of interest:
w <- attr(m.PHIHOX@model$opt.coef, "dimnames")[[1]]
nc <- as.numeric(sapply(w, function(x){gsub("[^0-9.]", "", x)}))
rn <- rank(nc)
if(any(duplicated(rn))){ warning("Duplicate bands detected") }
UPPER = c((nc[rn][-length(nc)]+nc[rn][-1])/2, nc[rn][length(nc)]+1)
wns <- data.frame(BAND=w, LOWER=c(nc[rn][1]-1, UPPER[-length(nc)]), UPPER=UPPER)
str(wns)
## compare with:
str(wavenumbers)
```

Index