

Package ‘soilDB’

November 17, 2017

Type Package

Title Soil Database Interface

Version 1.8.14

Date 2017-11-11

Author D.E. Beaudette and J. Skovlin, and S.M. Roecker

Maintainer D.E. Beaudette <dylan.beaudette@ca.usda.gov>

Description A collection of functions for reading data from USDA-NCSS soil databases.

License GPL (>= 3)

LazyLoad yes

Depends R (>= 3.0.0), aqp

Imports grDevices, graphics, stats, utils, plyr, Hmisc, xml2, sp,
reshape2, raster, curl

Suggests lattice, dismo, rgdal, jsonlite, RODBC, httr, rgeos, rvest

Repository CRAN

URL <http://ncss-tech.github.io/AQP/>

BugReports <https://github.com/ncss-tech/soilDB/issues>

RoxygenNote 6.0.1

NeedsCompilation no

Date/Publication 2017-11-16 23:00:28 UTC

R topics documented:

soilDB-package	2
fetchHenry	3
fetchKSSL	5
fetchNASIS	7
fetchNASISLabData	9
fetchOSD	10
fetchPedonPC	11

fetchRaCA	12
fetchSCAN	13
fetchSDA_component	14
get_colors_from_NASIS_db	17
get_colors_from_pedon_db	18
get_extended_data_from_pedon_db	18
get_hz_data_from_NASIS_db	19
get_hz_data_from_pedon_db	20
get_lablayer_data_from_NASIS_db	21
get_labpedon_data_from_NASIS_db	21
get_phlabresults_data_from_NASIS_db	22
get_site_data_from_NASIS_db	23
get_site_data_from_pedon_db	24
get_veg_from_AK_Site	25
get_veg_from_MT_veg_db	26
get_veg_from_NPS_PLOTS_db	27
get_veg_other_from_MT_veg_db	27
get_veg_species_from_MT_veg_db	28
gSSURGO.chunk	29
KSSL_VG_model	29
loafercreek	31
mapunit_geom_by_ll_bbox	34
parseWebReport	35
SDA_query	36
SDA_query_features	38
seriesExtent	39
simplifyColorData	40
SoilWeb_spatial_query	41
unicode	42
Index	43

soilDB-package	<i>Soil Database Interface</i>
----------------	--------------------------------

Description

This package provides methods for extracting soils information from local PedonPC and AK Site databases (MS Access format), local NASIS databases (MS SQL Server), and the SDA webservice. Currently USDA-NCSS data sources are supported, however, there are plans to develop interfaces to outside systems such as the Global Soil Mapping project.

Details

It can be difficult to locate all of the dependencies required for sending/processing SOAP requests, especially on UNIX-like operating systems. Windows binary packages for the dependencies can be found [here](#). See `fetchPedonPC` for a simple wrapper function that should suffice for typical site/pedon/hz queries. An introduction to the soilDB package can be found [here](#).

Author(s)

J.M. Skovlin and D.E. Beaudette

See Also

[fetchPedonPC](#), [fetchNASIS](#), [SDA_query](#), [loafercreek](#)

fetchHenry	<i>Download Data from the Henry Mount Soil Temperature and Water Database</i>
------------	---

Description

This function is a front-end to the REST query functionality of the Henry Mount Soil Temperature and Water Database.

Usage

```
fetchHenry(what='all', usersiteid = NULL, project = NULL, sso = NULL,
gran = "day", start.date = NULL, stop.date = NULL,
pad.missing.days = TRUE, soiltemp.summaries = TRUE)
```

Arguments

what	type of data to return: 'sensors': sensor metadata only 'soiltemp': sensor metadata + soil temperature data 'soilVWC': sensor metadata + soil moisture data 'airtemp': sensor metadata + air temperature data 'waterlevel': sensor metadata + water level data 'all': sensor metadata + all sensor data
usersiteid	(optional) filter results using a NASIS user site ID
project	(optional) filter results using a project ID
sso	(optional) filter results using a soil survey office code
gran	data granularity: "day", "week", "month", "year"; returned data are averages
start.date	(optional) starting date filter
stop.date	(optional) ending date filter
pad.missing.days	should missing data ("day" granularity) be filled with NA? see details
soiltemp.summaries	should soil temperature ("day" granularity only) be summarized? see details

Details

Filling missing days with NA is useful for computing and index of how complete the data are, and for estimating (mostly) unbiased MAST and seasonal mean soil temperatures. Summaries are computed by first averaging over Julian day, then averaging over all days of the year (MAST) or just those days that occur within "summer" or "winter". This approach makes it possible to estimate summaries in the presence of missing data. The quality of summaries should be weighted by the number of "functional years" (number of years with non-missing data after combining data by Julian day) and "complete years" (number of years of data with ≥ 365 days of non-missing data).

Value

a list containing:

sensors	a SpatialPointsDataFrame object containing site-level information
soiltemp	a data.frame object containing soil temperature timeseries data
soilVWC	a data.frame object containing soil moisture timeseries data
airtemp	a data.frame object containing air temperature timeseries data
waterlevel	a data.frame object containing water level timeseries data

Note

This function and the back-end database are very much a work in progress.

Author(s)

D.E. Beaudette

See Also

[fetchSCAN](#)

Examples

```
## Not run:
library(lattice)

# get CA630 data as daily averages
x <- fetchHenry(project='CA630', gran = 'day')

# inspect data gaps
levelplot(factor(!is.na(sensor_value)) ~ doy * factor(year) | name,
data=x$soiltemp, col.regions=c('grey', 'RoyalBlue'), cuts=1,
colorkey=FALSE, as.table=TRUE, scales=list(alternating=3),
par.strip.text=list(cex=0.75), strip=strip.custom(bg='yellow'),
xlab='Julian Day', ylab='Year')

## End(Not run)
```

 fetchKSSL

Fetch KSSL Data (EXPERIMENTAL)

Description

Get soil characterization and morphologic data via BBOX, MLRA, or series name query, from the KSSL database.

Usage

```
fetchKSSL(series=NULL, bbox=NULL, mlra=NULL, pedlabsampnum=NULL,
pedon_id=NULL, pedon_key=NULL, returnMorphologicData=FALSE)
```

Arguments

series	a single soil series name, case insensitive
bbox	a bounding box in WGS84 geographic coordinates e.g. c(-120, 37, -122, 38)
mlra	an MLRA ID, e.g. "18" or "22A"
pedlabsampnum	a single KSSL pedon lab sample number
pedon_id	a single user pedon ID
pedon_key	a single KSSL internal pedon ID
returnMorphologicData	optionally request basic morphologic data, see details section

Details

This is an experimental interface to a subset for the most commonly used data from a snapshot of KSSL (lab characterization) and NASIS (morphologic) data. The snapshots were last updated September 2017 (KSSL / NASIS).

Series-queries are case insensitive. Series name is based on the "correlated as" field (from KSSL snapshot) when present. The "sampled as" classification was promoted to "correlated as" if the "correlated as" classification was missing.

When returnMorphologicData is TRUE, the resulting object is a list. The standard output from fetchKSSL (SoilProfileCollection object) is stored in the named element "SPC". The additional elements are basic morphologic data: horizon colors, rock fragments, pores, and structure. There is a 1:many relationship between the horizon data in "SPC" and the additional dataframes in morph. See examples for ideas on how to "flatten" these tables.

Function arguments (series, mlra, etc.) are NOT vectorized: the first element of a vector will be used when supplied as a filter. See the [fetchKSSL tutorial](#) for ideas on how to iterate over a set of IDs.)

Value

a SoilProfileCollection object when returnMorphologicData is FALSE, otherwise a list.

Note

SoilWeb maintains a snapshot of these KSSL and NASIS data. The SoilWeb snapshot was developed using methods described here: <https://github.com/dylanbeaudette/process-kssl-snapshot>. Please use the link below for the live data.

Author(s)

D.E. Beaudette

References

<http://ncsslslabdatamart.sc.egov.usda.gov/>

See Also

[fetchOSD](#)

Examples

```
## Not run:
# search by series name
s <- fetchKSSL(series='auburn')

# search by bounding-box
# s <- fetchKSSL(bbox=c(-120, 37, -122, 38))

# how many pedons
length(s)

# plot
par(mar=c(0,0,0,0))
plot(s, name='hzn_desgn', max.depth=150)

# get morphologic data too
library(soilDB)
library(plyr)
library(reshape2)

# get lab and morphologic data
s <- fetchKSSL(series='auburn', returnMorphologicData = TRUE)

# extract SPC
pedons <- s$SPC

# simplify color data
s.colors <- simplifyColorData(s$morph$phcolor, id.var = 'labsampnum')

# merge color data into SPC
h <- horizons(pedons)
h <- join(h, s.colors, by='labsampnum', type='left', match='first')
```

```
horizons(pedons) <- h

# check
par(mar=c(0,0,0,0))
plot(pedons, color='moist_soil_color', print.id=FALSE)

# simplify fragment data
s.f frags <- simplifyFragmentData(s$morph$phfrags, id.var='labsampnum')

# merge fragment data into SPC
h <- horizons(pedons)
h <- join(h, s.f frags, by='labsampnum', type='left', match='first')
horizons(pedons) <- h

# check
par(mar=c(0,0,3,0))
plot(pedons, color='total_frags_pct', print.id=FALSE)

## End(Not run)
```

fetchNASIS	<i>Fetch commonly used site/pedon/horizon data from a PedonPC database.</i>
------------	---

Description

Fetch commonly used site/pedon/horizon data from a PedonPC or local NASIS database, return as a SoilProfileCollection object.

Usage

```
fetchNASIS(rmHzErrors = TRUE, nullFragsAreZero=TRUE, soilColorState='moist', lab='FALSE')
fetchNASIS_component_data(rmHzErrors=TRUE, fill = FALSE)
getHzErrorsNASIS(strict=TRUE)
get_comonth_from_NASIS_db(fill=FALSE)
```

Arguments

rmHzErrors	should pedons with horizonation errors be removed from the results? (default: TRUE)
nullFragsAreZero	should fragment volumes of NULL be interpreted as 0? (default: TRUE), see details
soilColorState	which colors should be used to generate the convenience field 'soil_color'? ('moist' 'dry')

lab	should the phlabresults child table be fetched (default: FALSE)
strict	should horization by strictly enforced? (TRUE)
fill	should missing "month" rows in the comonth table be filled with NA (FALSE)

Details

The value of `nullFragmentsAreZero` will have a significant impact on the rock fragment fractions returned by `fetchNASIS`. Set `nullFragmentsAreZero = FALSE` in those cases where there are many data-gaps and NULL rock fragment values should be interpreted as NULLs. Set `nullFragmentsAreZero = TRUE` in those cases where NULL rock fragment values should be interpreted as 0.

This function attempts to do most of the boilerplate work when extracting site/pedon/horizon data from a local NASIS database. Pedons that are missing horizon data, or have errors in their horization are excluded from the returned object, however, their IDs are printed on the console. Pedons with combination horizons (e.g. B/C) are erroneously marked as errors due to the way in which they are stored in NASIS as two overlapping horizon records.

See [getHzErrorsNASIS](#) for a simple approach to identifying pedons with problematic horization.

See the [NASIS component tutorial](#), and [NASIS pedon tutorial](#) for more information.

Value

a `SoilProfileCollection` class object

Note

This function currently works only on Windows, and requires a 'nasis_local' ODBC connection.

Author(s)

D. E. Beaudette and J. M. Skovlin

Examples

```
## Not run:
# query depends on some pedon data, queried against the national database
# note that you must setup this connection ahead of time
f <- fetchNASIS()

# plot only those profiles with densic contact
plot(f[which(f$densic.contact), ], name='hzname')

# get basic component data from local NASIS, after performing a
# DMU-* query against the national database
fc <- fetchNASIS_component_data()

## End(Not run)
```

fetchNASISLabData *Fetch lab data used site/horizon data from a PedonPC database.*

Description

Fetch KSSL laboratory pedon/horizon layer data from a local NASIS database, return as a SoilProfileCollection object.

Usage

```
fetchNASISLabData()
```

Details

This function currently works only on Windows, and requires a 'nasis_local' ODBC connection.

Value

a SoilProfileCollection class object

Note

This function attempts to do most of the boilerplate work when extracting KSSL laboratory site/horizon data from a local NASIS database. Lab pedons that have errors in their horizonation are excluded from the returned object, however, their IDs are printed on the console. See [getHzErrorsNASIS](#) for a simple approach to identifying pedons with problematic horizonation.

Author(s)

J.M. Skovlin and D.E. Beaudette

See Also

[get_labpedon_data_from_NASIS_db](#)

Examples

```
## Not run:
# query depends on some lab data, queried against the national database
# note that you must setup this connection ahead of time
# see inst/doc/setup_ODBC_local_NASIS.pdf
f <- fetchNASISLabData()

# plot only those profiles with densic contact
#plot(f[which(f$densic.contact), ], name='hzname')

## End(Not run)
```

`fetchOSD`*Fetch Official Series Description (OSD) Data*

Description

This functions fetches a limited subset of horizon and site-level attributes for named soil series, from the SoilWeb system.

Usage

```
fetchOSD(soils, colorState = 'moist')
```

Arguments

<code>soils</code>	a character vector of named soil series
<code>colorState</code>	color state for horizon soil color visualization: "moist" or "dry"

Details

the search is case-insensitive

Value

a `SoilProfileCollection` object containing basic soil morphology and taxonomic information.

Note

SoilWeb maintains a snapshot of the Official Series Description data. Please use the link above for the live data.

Author(s)

D.E. Beaudette

References

http://www.nrcs.usda.gov/wps/portal/nrcs/detailfull/soils/home/?cid=nrcs142p2_053587

Examples

```
## Not run:
# soils of interest
s.list <- c('musick', 'cecil', 'drummer', 'amador', 'pentz',
'reiff', 'san joaquin', 'montpellier', 'grangeville', 'pollasky', 'ramona')

# fetch and convert data into an SPC
s.moist <- fetchOSD(s.list, colorState='moist')
s.dry <- fetchOSD(s.list, colorState='dry')
```

```
# plot profiles
# moist soil colors
par(mar=c(0,0,0,0), mfrow=c(2,1))
plot(s.moist, name='hzname', cex.names=0.85, axis.line.offset=-4)
plot(s.dry, name='hzname', cex.names=0.85, axis.line.offset=-4)

## End(Not run)
```

`fetchPedonPC`*Fetch commonly used site/horizon data from a PedonPC v.5 database.*

Description

Fetch commonly used site/horizon data from a version 5.x PedonPC database, return as a SoilProfileCollection object.

Usage

```
fetchPedonPC(dsn)
getHzErrorsPedonPC(dsn, strict=TRUE)
```

Arguments

<code>dsn</code>	The path to a PedonPC version 5.x database
<code>strict</code>	should horizonation by strictly enforced? (TRUE)

Details

This function currently works only on Windows.

Value

a SoilProfileCollection class object

Note

This function attempts to do most of the boilerplate work when extracting site/horizon data from a PedonPC or local NASIS database. Pedons that have errors in their horizonation are excluded from the returned object, however, their IDs are printed on the console. See [getHzErrorsPedonPC](#) for a simple approach to identifying pedons with problematic horizonation. Records from the 'taxhistory' table are selected based on 1) most recent record, or 2) record with the least amount of missing data.

Author(s)

D. E. Beaudette and J. M. Skovlin

See Also

[get_hz_data_from_pedon_db](#)

Examples

```
## Not run:
# path to local PedonPC back-end DB
dsn <- "S:/Service_Center/NRCS/pedon/pedon.accdb"

# get routinely used soil data SoilProfileCollection object
f <- fetchPedonPC(dsn)

# plot only those profiles with densic contact
plot(f[which(f$densic.contact), ], name='hzname')

## End(Not run)
```

 fetchRaCA

Fetch KSSL Data (EXPERIMENTAL)

Description

Get Rapid Carbon Assessment (RaCA) data via state, geographic bounding-box, RaCA site ID, or series query from the SoilWeb system.

Usage

```
fetchRaCA(series = NULL, bbox = NULL, state = NULL, rcasiteid = NULL, get.vnir = FALSE)
```

Arguments

series	a soil series name, case insensitive
bbox	a bounding box in WGS84 geographic coordinates e.g. c(-120, 37, -122, 38), constrained to a 5-degree block
state	a two-letter US state abbreviation, case insensitive
rcasiteid	an RaCA site id (e.g. 'C1609C01')
get.vnir	boolean, should associated VNIR spectra be downloaded? (see details)

Details

The VNIR spectra associated with RaCA data are quite large [each gzip-compressed VNIR spectra record is about 6.6kb], so requests for these data are disabled by default. Note that VNIR spectra can only be queried by soil series or geographic BBOX.

Value

pedons: a SoilProfileCollection object containing site/pedon/horizon data
 trees: a data.frame object containing tree DBH and height
 veg: a data.frame object containing plant species
 stock: a data.frame object containing carbon quantities (stocks) at standardized depths
 sample: a data.frame object containing sample-level bulk density and soil organic carbon values
 spectra: a numeric matrix containing VNIR reflectance spectra from 350–2500 nm

Author(s)

D.E. Beaudette, USDA-NRCS staff

References

http://www.nrcs.usda.gov/wps/portal/nrcs/detail/soils/survey/?cid=nrcs142p2_054164
[fetchRaCA\(\) Tutorial](#)

See Also

[fetchOSD](#)

Examples

```
## Not run:
# search by series name
s <- fetchRaCA(series='auburn')

# search by bounding-box
# s <- fetchRaCA(bbox=c(-120, 37, -122, 38))

# check structure
str(s, 1)

# extract pedons
p <- s$pedons

# how many pedons
length(p)

# plot
par(mar=c(0,0,0,0))
plot(p, name='hzn_desgn', max.depth=150)

## End(Not run)
```

fetchSCAN

Fetch SCAN Data

Description

Query soil/climate data from USDA-NRCS SCAN Stations (experimental)

Usage

```
# get SCAN data
fetchSCAN(site.code, year, report='SCAN', req=NULL)
# get SCAN station metadata
SCAN_sensor_metadata(site.code)
```

Arguments

site.code a vector of site codes
year a vector of years
report report name, single value only
req list of SCAN request parameters, for backwards-compatibility only

Details

See [The fetchSCAN tutorial for details](#). These functions require the 'httr' and 'rvest' libraries.

Value

a data.frame object

Note

SCAN_sensor_metadata() is known to crash on 32bit R / libraries (Windows).

Author(s)

D.E. Beaudette

References

<https://www.wcc.nrcs.usda.gov/index.html>

Examples

```
## Not run:  
# get data: new interface  
x <- fetchSCAN(site.code=c(356, 2072), year=c(2015, 2016))  
str(x)  
  
# get sensor metadata  
m <- SCAN_sensor_metadata(site.code=c(356, 2072))  
  
## End(Not run)
```

fetchSDA_component	<i>Extract component tables from a local NASIS Database or Soil Data Access</i>
--------------------	---

Description

Get, format, impute, and return component tables.

Usage

```

fetchSDA_component(WHERE = NULL, duplicates = FALSE, rmHzErrors = FALSE)
fetchLIMS_component(projectname, rmHzErrors = FALSE, fill = FALSE)
get_mapunit_from_SDA(WHERE = NULL)
get_mapunit_from_LIMS(projectname)
get_component_from_SDA(WHERE = NULL, duplicates = FALSE)
get_component_from_LIMS(projectname)
get_chorizon_from_SDA(WHERE = NULL, duplicates = FALSE)
get_chorizon_from_LIMS(projectname, fill = FALSE)
get_cosoilmoist_from_SDA(WHERE = NULL, duplicates = FALSE, impute = TRUE)
get_cosoilmoist_from_LIMS(projectname, impute = TRUE)
get_cosoilmoist_from_NASIS(impute = TRUE)

```

Arguments

WHERE	text string formatted as an SQL WHERE clause (default: FALSE)
projectname	text string value identifying the project name inserted into a SQL WHERE clause (default: NA)
duplicates	logical; if TRUE duplicate nationalmusym are returned
impute	replace missing (i.e. NULL) values with "Not_Populated" for categorical data, or the "RV" for numeric data or 201 cm if the "RV" is also NULL (default: TRUE)
rmHzErrors	should pedons with horizonation errors be removed from the results? (default: FALSE)
fill	should rows with missing component ids be removed NA (FALSE)

Details

The SDA and LIMS functions can get and fetch data with an internet connection and a WHERE clause.

If the duplicates argument is set to TRUE, duplicate components are returned. This is not necessary with data returned from NASIS, which has one unique national map unit. SDA has duplicate map national map units, one for each legend it exists in.

The function `get_cosoilmoist_from_NASIS_db()` only works only on Windows , and requires a 'nasis_local' ODBC connection. See the [NASIS ODBC Setup tutorial](#) for instructions.

Value

A dataframe or list with the results.

Author(s)

Stephen Roecker

Examples

```

## Not run:
library(soilDB)
library(ggplot2)
library(gridExtra)

# query soil components by areasympol and musym
test = fetchSDA_component(WHERE = "areasympol = 'IN005' AND musym = 'MnpB2'")
test = test$spc

# profile plot
plot(test)

# convert the data for depth plot
clay_slice = horizons(slice(test, 0:200 ~ claytotal_l + claytotal_r + claytotal_h))
names(clay_slice) <- gsub("claytotal_", "", names(clay_slice))

om_slice = horizons(slice(test, 0:200 ~ om_l + om_r + om_h))
names(om_slice) = gsub("om_", "", names(om_slice))

test2 = rbind(data.frame(clay_slice, var = "clay"),
              data.frame(om_slice, var = "om")
              )

h = merge(test2, site(test)[c("nationalmusym", "cokey", "compname", "compct_r")],
          by = "cokey",
          all.x = TRUE
          )

# depth plot of clay content by soil component
gg_comp <- function(x) {
  ggplot(x) +
    geom_line(aes(y = r, x = hzdept_r)) +
    geom_line(aes(y = r, x = hzdept_r)) +
    geom_ribbon(aes(ymin = 1, ymax = h, x = hzdept_r), alpha = 0.2) +
    xlim(200, 0) +
    xlab("depth (cm)") +
    facet_grid(var ~ nationalmusym + paste(compname, compct_r)) +
    coord_flip()
}
g1 <- gg_comp(subset(h, var == "clay"))
g2 <- gg_comp(subset(h, var == "om"))

grid.arrange(g1, g2)

# query cosoilmoist (e.g. water table data) by mukey
# NA depths are interpreted as (???) with impute=TRUE argument
x <- get_cosoilmoist_from_SDA(WHERE = "mukey = '1395352'", impute = TRUE)

ggplot(x, aes(x = as.integer(month), y = dept_r, lty = status)) +
  geom_rect(aes(xmin = as.integer(month), xmax = as.integer(month) + 1,

```

```
      ymin = 0, ymax = max(x$depb_r),
      fill = flodfreqcl)) +
geom_line(cex = 1) +
geom_point() +
geom_ribbon(aes(ymin = dept_l, ymax = dept_h), alpha = 0.2) +
ylim(max(x$depb_r), 0) +
xlab("month") + ylab("depth (cm)") +
scale_x_continuous(breaks = 1:12, labels = month.abb, name="Month") +
facet_wrap(~ paste0(compname, ' (', comppct_r, ')')) +
ggtitle(paste0(x$nationalmusym[1],
': Water Table Levels from Component Soil Moisture Month Data'))

## End(Not run)
```

get_colors_from_NASIS_db

Extract Soil Color Data from a local NASIS Database

Description

Get, format, mix, and return color data from a NASIS database.

Usage

```
get_colors_from_NASIS_db()
```

Details

This function currently works only on Windows.

Value

A dataframe with the results.

Author(s)

Jay M. Skovlin and Dylan E. Beaudette

See Also

[simplifyColorData](#), [get_hz_data_from_NASIS_db](#), [get_site_data_from_NASIS_db](#)

`get_colors_from_pedon_db`*Extract Soil Color Data from a PedonPC Database*

Description

Get, format, mix, and return color data from a PedonPC database.

Usage

```
get_colors_from_pedon_db(dsn)
```

Arguments

`dsn` The path to a 'pedon.mdb' database.

Details

This function currently works only on Windows.

Value

A dataframe with the results.

Author(s)

Dylan E. Beaudette and Jay M. Skovlin

See Also

[get_hz_data_from_pedon_db](#), [get_site_data_from_pedon_db](#)

`get_extended_data_from_pedon_db`*Extract accessory tables and summaries from a local pedonPC Database*

Description

Extract accessory tables and summaries from a local pedonPC Database.

Usage

```
get_extended_data_from_pedon_db(dsn)
```

Arguments

`dsn` The path to a 'pedon.mdb' database.

Details

This function currently works only on Windows.

Value

A list with the results.

Author(s)

Jay M. Skovlin and Dylan E. Beaudette

See Also

[get_hz_data_from_pedon_db](#), [get_site_data_from_pedon_db](#)

`get_hz_data_from_NASIS_db`

Extract Horizon Data from a local NASIS Database

Description

Get horizon-level data from a local NASIS database.

Usage

```
get_hz_data_from_NASIS_db()
```

Details

This function currently works only on Windows.

Value

A dataframe.

Note

NULL total rock fragment values are assumed to represent an `_absense_` of rock fragments, and set to 0.

Author(s)

Jay M. Skovlin and Dylan E. Beaudette

See Also

[get_hz_data_from_NASIS_db](#), [get_site_data_from_NASIS_db](#)

`get_hz_data_from_pedon_db`

Extract Horizon Data from a PedonPC Database

Description

Get horizon-level data from a PedonPC database.

Usage

```
get_hz_data_from_pedon_db(dsn)
```

Arguments

`dsn` The path to a 'pedon.mdb' database.

Details

This function currently works only on Windows.

Value

A dataframe.

Note

NULL total rock fragment values are assumed to represent an `_absense_` of rock fragments, and set to 0.

Author(s)

Dylan E. Beaudette and Jay M. Skovlin

See Also

[get_colors_from_pedon_db](#), [get_site_data_from_pedon_db](#)

`get_lablayer_data_from_NASIS_db`

Extract lab pedon layer data from a local NASIS Database

Description

Get lab pedon layer-level(horizon-level) data from a local NASIS database.

Usage

`get_lablayer_data_from_NASIS_db()`

Details

This function currently works only on Windows, and requires a 'nasis_local' ODBC connection.

Value

A dataframe.

Note

This function queries KSSL laboratory site/horizon data from a local NASIS database from the lab layer data table.

Author(s)

Jay M. Skovlin and Dylan E. Beaudette

See Also

[get_labpedon_data_from_NASIS_db](#)

`get_labpedon_data_from_NASIS_db`

Extract lab pedon data from a local NASIS Database

Description

Get lab pedon-level data from a local NASIS database.

Usage

`get_labpedon_data_from_NASIS_db()`

Details

This function currently works only on Windows, and requires a 'nasis_local' ODBC connection.

Value

A dataframe.

Note

This function queries KSSL laboratory site/horizon data from a local NASIS database from the lab pedon data table.

Author(s)

Jay M. Skovlin and Dylan E. Beaudette

See Also

[get_lablayer_data_from_NASIS_db](#)

`get_phlabresults_data_from_NASIS_db`

Extract phlabresults table from a local NASIS Database

Description

Get, format, aggregate, and return phlabresults table from a NASIS database.

Usage

```
get_phlabresults_data_from_NASIS_db()
```

Details

This function currently works only on Windows.

Value

A dataframe with the results.

Author(s)

Stephen Roecker

See Also

[get_hz_data_from_NASIS_db](#), [get_site_data_from_NASIS_db](#)

`get_site_data_from_NASIS_db`*Extract Site Data from a local NASIS Database*

Description

Get site-level data from a local NASIS database.

Usage

```
get_site_data_from_NASIS_db()
```

Details

When multiple "site bedrock" entries are present, only the shallowest is returned by this function.

Value

A dataframe.

Note

This function currently works only on Windows.

Author(s)

Jay M. Skovlin and Dylan E. Beaudette

See Also

[get_hz_data_from_NASIS_db](#),

Examples

```
## Not run:

## Example: export / convert DMS coordinates from NASIS and save to DD import file

# load required libraries
library(soilDB)
library(rgdal)
library(plyr)

# get site data from NASIS
s <- get_site_data_from_NASIS_db()

# keep only those pedons with real coordinates
good.idx <- which(!is.na(s$x))
```

```

s <- s[good.idx, ]

# investigate multiple datums:
table(s$datum, useNA='always')

## this is not universally appropriate!
# assume missing is NAD83
s$datum[is.na(s$datum)] <- 'NAD83'

# check: OK
table(s$datum, useNA='always')

# convert to NAD83
old.coords <- cbind(s$x, s$y)

# add temp colum for projection information, and fill with proj4 style info
s$proj4 <- rep(NA, times=nrow(s))
s$proj4 <- paste('+proj=longlat +datum=', s$datum, sep='')

# iterate over pedons, and convert to WGS84
new.coords <- ddply(s, 'peiid',
  .progress='text', .fun=function(i) {
    coordinates(i) <- ~ x + y
    proj4string(i) <- CRS(i$proj4)
    i.t <- spTransform(i, CRS('+proj=longlat +datum=WGS84'))
    i.c <- as.matrix(coordinates(i.t))
    return(data.frame(x.new=i.c[, 1], y.new=i.c[, 2]))
  })

# merge in new coordinates
s <- join(s, new.coords)

# any changes?
summary(sqrt(apply((s[, c('x', 'y')] - s[, c('x.new', 'y.new')])^2, 1, sum)))

# save to update file for use with "Import of Standard WGS84 Georeference" calculation in NASIS
# note that this defines the coordinate source as "GPS", hence the last column of '1's.
std.coordinates.update.data <- unique(cbind(s[, c('siteiid', 'y.new', 'x.new')], 1))
# save to file
write.table(std.coordinates.update.data,
file='c:/data/sgeoref.txt', col.names=FALSE, row.names=FALSE, sep='|')

## End(Not run)

```

```
get_site_data_from_pedon_db
```

Extract Site Data from a PedonPC Database

Description

Get site-level data from a PedonPC database.

Usage

```
get_site_data_from_pedon_db(dsn)
```

Arguments

dsn The path to a 'pedon.mdb' database.

Value

A dataframe.

Note

This function currently works only on Windows.

Author(s)

Dylan E. Beaudette and Jay M. Skovlin

See Also

[get_hz_data_from_pedon_db](#), [get_veg_from_AK_Site](#),

`get_veg_from_AK_Site` *Retrieve Vegetation Data from an AK Site Database*

Description

Retrieve Vegetation Data from an AK Site Database

Usage

```
get_veg_from_AK_Site(dsn)
```

Arguments

dsn file path the the AK Site access database

Value

A dataframe with vegetation data in long format, linked to site ID.

Note

This function currently works only on Windows.

Author(s)

Dylan E. Beaudette

See Also

[get_hz_data_from_pedon_db](#), [get_site_data_from_pedon_db](#)

`get_veg_from_MT_veg_db`

Extract Site and Plot-level Data from a Montana RangeDB database

Description

Get Site and Plot-level data from a Montana RangeDB database.

Usage

```
get_veg_from_MT_veg_db(dsn)
```

Arguments

`dsn` The name of the Montana RangeDB front-end database connection (see details).

Details

This function currently works only on Windows.

Value

A dataframe.

Author(s)

Jay M. Skovlin

See Also

[get_veg_species_from_MT_veg_db](#), [get_veg_other_from_MT_veg_db](#)

`get_veg_from_NPS_PLOTS_db`*Retrieve Vegetation Data from an NPS PLOTS Database*

Description

Used to extract species, stratum, and cover vegetation data from a backend NPS PLOTS Database. Currently works for any Microsoft Access database with an .mdb file format.

Usage`get_veg_from_NPS_PLOTS_db(dsn)`**Arguments**

dsn file path to the NPS PLOTS access database on your system.

Value

A dataframe with vegetation data in a long format with linkage to NRCS soil pedon data via the site_id key field.

Note

This function currently only works on Windows.

Author(s)

Jay M. Skovlin

`get_veg_other_from_MT_veg_db`*Extract cover composition data from a Montana RangeDB database*

Description

Get cover composition data from a Montana RangeDB database.

Usage`get_veg_other_from_MT_veg_db(dsn)`**Arguments**

dsn The name of the Montana RangeDB front-end database connection (see details).

Details

This function currently works only on Windows.

Value

A dataframe.

Author(s)

Jay M. Skovlin

See Also

[get_veg_from_MT_veg_db](#), [get_veg_species_from_MT_veg_db](#)

`get_veg_species_from_MT_veg_db`

Extract species-level Data from a Montana RangeDB database

Description

Get species-level data from a Montana RangeDB database.

Usage

```
get_veg_species_from_MT_veg_db(dsn)
```

Arguments

`dsn` The name of the Montana RangeDB front-end database connection (see details).

Details

This function currently works only on Windows.

Value

A dataframe.

Author(s)

Jay M. Skovlin

See Also

[get_veg_from_MT_veg_db](#), [get_veg_other_from_MT_veg_db](#)

gSSURGO.chunk

Gridded SSURGO Chunk

Description

A chunk of the gridded SSURGO database (gSSURGO)

Usage

```
data(gSSURGO.chunk)
```

Details

This is a 106x137 grid, cropped from the gSSURGO database. Cell values are map unit keys (mukey), stored as integers. No raster attribute table (RAT) is included with this sample data set. Note that this sample of the gSSURGO data has been modified such that cell values are map unit keys, rather than the gSSURGO integer key.

Source

http://www.nrcs.usda.gov/wps/portal/nrcs/detail//?cid=nrcs142p2_053628

Examples

```
## Not run:
data(gSSURGO.chunk)

## End(Not run)
```

KSSL_VG_model

Develop a Water Retention Curve from KSSL Data

Description

Water retention curve modeling via van Genuchten model and KSSL data.

Usage

```
KSSL_VG_model(VG_params, phi_min = 10^-6, phi_max = 10^8, pts = 100)
```

Arguments

VG_params	a data.frame or list object with the parameters of the van Genuchten model, see details
phi_min	lower limit for water potential in KPa
phi_max	upper limit for water potential in KPa
pts	number of points to include in estimated water retention curve

Details

This function was developed to work with measured or estimated parameters of the **van Genuchten model**, as generated by the **Rosetta model**. As such, VG_params should have the following format and conventions:

theta_r saturated water content, values should be in the range of {0, 1}

theta_s residual water content, values should be in the range of {0, 1}

alpha related to the inverse of the air entry suction, function expects log10-transformed values with units of cm

npar index of pore size distribution, function expects log10-transformed values with units of 1/cm

Value

A list with the following components:

VG_curve estimated water retention curve: paired estimates of water potential and water content

VG_inverse_function function for converting measured water content (theta, units of percent, range: {0, 1}) to estimated water potential (phi, units of KPa)

Note

A practical example is given in the **fetchSCAN tutorial**.

Author(s)

D.E. Beaudette

References

water retention curve estimation

van Genuchten, M.Th. (1980). "A closed-form equation for predicting the hydraulic conductivity of unsaturated soils". Soil Science Society of America Journal. 44 (5): 892-898.

Examples

```
# basic example
d <- data.frame(theta_r=0.0337216,
theta_s=0.4864061,
alpha=-1.581517,
npar=0.1227247)

vg <- KSSL_VG_model(d)

str(vg)
```

loafercreek

*LoaferCreek Soils***Description**

A SoilProfileCollection Object of Loafercreek Soils from the CA630 Soil Survey Area.

Usage

```
data(loafercreek)
```

Format

```
Formal class 'SoilProfileCollection' [package "aqp"] with 7 slots
..@ idcol      : chr "pedon_id"
..@ depthcols  : chr [1:2] "hzdept" "hzdepb"
..@ metadata   : 'data.frame': 1 obs. of 1 variable:
.. ..$ depth_units: chr "cm"
..@ horizons   : 'data.frame': 308 obs. of 29 variables:
.. ..$ phiid      : int [1:308] 1258320 1258321 1258322 1258323 1258324 1258325 2068689 2068690 20
.. ..$ pedon_id   : chr [1:308] "07SKC003" "07SKC003" "07SKC003" "07SKC003" ...
.. ..$ hzname     : chr [1:308] "A" "Bt1" "Bt2" "2Bt3" ...
.. ..$ hzdept    : int [1:308] 0 10 23 33 48 64 0 1 4 12 ...
.. ..$ hzdepb    : int [1:308] 10 23 33 48 64 64 1 4 12 28 ...
.. ..$ clay      : num [1:308] 15 19 32 38 28 NA NA 16 18 25 ...
.. ..$ silt      : num [1:308] 50 61 55 49 NA NA NA NA NA NA ...
.. ..$ sand      : num [1:308] 35 20 13 13 NA NA NA NA NA NA ...
.. ..$ texture_class : chr [1:308] "1" "sil" "sic1" "sic1" ...
.. ..$ phfield   : num [1:308] 6.8 6.7 6.9 6.9 6.7 ...
.. ..$ effervescence : chr [1:308] NA NA NA NA ...
.. ..$ labsampnum : chr [1:308] NA NA NA NA ...
.. ..$ total_frgs_pct: int [1:308] 10 24 5 10 23 0 0 10 10 18 ...
.. ..$ d_r       : num [1:308] 0.525 0.637 0.541 0.525 0.616 ...
.. ..$ d_g       : num [1:308] 0.34 0.441 0.331 0.34 0.452 ...
.. ..$ d_b       : num [1:308] 0.158 0.248 0.185 0.158 0.218 ...
.. ..$ m_r       : num [1:308] 0.399 0.525 0.541 0.525 0.525 ...
.. ..$ m_g       : num [1:308] 0.251 0.34 0.331 0.34 0.34 ...
.. ..$ m_b       : num [1:308] 0.113 0.158 0.185 0.158 0.158 ...
.. ..$ taxon_kind : chr [1:308] "series" "series" "series" "series" ...
.. ..$ soil_color : chr [1:308] "#66401D" "#865728" "#8A542F" "#865728" ...
.. ..$ gravel    : num [1:308] 10 16 5 10 0 NA NA 10 10 18 ...
.. ..$ paragravel : num [1:308] 0 0 0 0 18 NA NA 0 0 0 ...
.. ..$ cobbles   : num [1:308] 0 0 0 0 0 NA NA 0 0 0 ...
.. ..$ paracobbles : num [1:308] 0 8 0 0 5 NA NA 0 0 0 ...
.. ..$ stones    : num [1:308] 0 0 0 0 0 NA NA 0 0 0 ...
.. ..$ boulders  : num [1:308] 0 0 0 0 0 NA NA 0 0 0 ...
.. ..$ channers  : num [1:308] 0 0 0 0 0 NA NA 0 0 0 ...
```

```

.. ..$ flagstones      : num [1:308] 0 0 0 0 0 NA NA 0 0 0 ...
..@ site              : 'data.frame': 52 obs. of  46 variables:
.. ..$ pedon_id       : chr [1:52] "07SKC003" "09BAH008" "09CKS001" "09CKS006" ..
.. ..$ peiid          : int [1:52] 268820 493742 342445 374201 374205 374216 374219
.. ..$ site_id        : chr [1:52] "07CA630SKC003" "09CA630BAH008" "09CA630CKS001"
.. ..$ siteiid        : int [1:52] 269617 501286 342696 374485 374489 374500 374503
.. ..$ sampled_as     : chr [1:52] "Motherlode" "Loafercreek" "Loafercreek active"
.. ..$ correlated_as  : chr [1:52] "LOAFERCREEK" "Loafercreek" "Loafercreek" "Loa
.. ..$ hillslope_pos  : chr [1:52] "Backslope" "Footslope" "Backslope" "Backslope
.. ..$ datum          : chr [1:52] "NAD83" "NAD83" "NAD83" "NAD83" ...
.. ..$ elev           : num [1:52] 328 384 1014 210 202 ...
.. ..$ slope          : num [1:52] 9 3 22 29 4 36 50 18 22 32 ...
.. ..$ aspect         : int [1:52] 344 102 220 235 115 185 185 335 45 52 ...
.. ..$ plantassocnm   : chr [1:52] "Blue Oak Woodland" "Blue Oak Woodland/Grasslan
.. ..$ bedrckdepth    : int [1:52] 64 68 110 86 96 72 93 65 72 70 ...
.. ..$ bedrock_kind   : chr [1:52] "Greenstone" "Metavolcanics" "Greenstone" "Gree
.. ..$ bedrock_hardness : chr [1:52] NA "Noncemented" NA NA ...
.. ..$ describer      : chr [1:52] "Stacy Kavanaugh" "Bev Harben" "Chris Savastio"
.. ..$ psctopdepth    : int [1:52] 23 12 9 3 2 12 8 5 6 25 ...
.. ..$ pscbotdepth    : int [1:52] 64 62 59 53 52 57 58 55 56 70 ...
.. ..$ part_size_class : chr [1:52] "fine-loamy" "fine-loamy" "fine-loamy" "fine-l
.. ..$ tax_subgroup    : chr [1:52] "Ultic Haploxeralfs" "Ultic Haploxeralfs" "Ulti
.. ..$ obs_date        : POSIXct[1:52], format: "2007-03-21" "2009-06-03" "2009-03-
.. ..$ pedon_purpose     : chr [1:52] "full pedon description" "full pedon descriptio
.. ..$ pedon_type      : chr [1:52] "within range of series" "within range of series
.. ..$ pedlabsampnum   : chr [1:52] NA NA "S09CA009002" NA ...
.. ..$ ochric.epipedon : logi [1:52] TRUE TRUE TRUE TRUE TRUE TRUE ...
.. ..$ cambic.horizon  : logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ paralithic.contact : logi [1:52] TRUE TRUE TRUE TRUE TRUE TRUE ...
.. ..$ lithic.contact  : logi [1:52] FALSE FALSE TRUE FALSE FALSE FALSE ...
.. ..$ mollic.epipedon : logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ argillic.horizon : logi [1:52] TRUE TRUE TRUE TRUE TRUE TRUE ...
.. ..$ umbric.epipedon : logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ paralithic.materials : logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ lithologic.discontinuity : logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ andic.soil.properties : logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ densic.contact  : logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ aquic.conditions : logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ duripan         : logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ slickensides    : logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ kandic.horizon  : logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ redox.depletions.with.chroma.2.or.less: logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ reduced.matrix  : logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ abrupt.textural.change : logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ calcic.horizon  : logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ redox.concentrations : logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ limnic.material : logi [1:52] FALSE FALSE FALSE FALSE FALSE FALSE ...
.. ..$ musym           : chr [1:52] "7086" "7085" "7086" "7086" ...

```

```

..@ sp          :Formal class 'SpatialPoints' [package "sp"] with 3 slots
.. .. ..@ coords      : num [1:52, 1:2] 723623 716141 713132 712414 712479 ...
.. .. ..- attr(*, "dimnames")=List of 2
.. .. .. ..$ : chr [1:52] "248" "406" "495" "500" ...
.. .. .. ..$ : chr [1:2] "x" "y"
.. .. ..@ bbox        : num [1:2, 1:2] 694821 4179313 729963 4233623
.. .. ..- attr(*, "dimnames")=List of 2
.. .. .. ..$ : chr [1:2] "x" "y"
.. .. .. ..$ : chr [1:2] "min" "max"
.. .. ..@ proj4string:Formal class 'CRS' [package "sp"] with 1 slots
.. .. .. ..@ projargs: chr " +proj=utm +zone=10 +ellps=GRS80 +datum=NAD83 +units=m +no_defs +towg
..@ diagnostic:'data.frame': 170 obs. of  8 variables:
.. ..$ pedon_id      : chr [1:170] "07SKC003" "07SKC003" "07SKC003" "09BAH008" ...
.. ..$ peiid         : int [1:170] 268820 268820 268820 493742 493742 493742 342445 342445 342445 34244
.. ..$ musym         : chr [1:170] "7086" "7086" "7086" "7085" ...
.. ..$ correlated_as: chr [1:170] "LOAFERCREEK" "LOAFERCREEK" "LOAFERCREEK" "Loafercreek" ...
.. ..$ pedon_id      : chr [1:170] "07SKC003" "07SKC003" "07SKC003" "09BAH008" ...
.. ..$ diag_kind     : chr [1:170] "ochric epipedon" "argillic horizon" "paralithic contact" "ochric e
.. ..$ featdept      : int [1:170] 0 23 64 1 12 68 0 9 97 110 ...
.. ..$ featdepb      : int [1:170] 23 64 NA 12 68 100 9 97 110 NA ...

```

Examples

```

## Not run:
# load example dataset
data(gopheridge)

# what kind of object is this?
class(gopheridge)

# tighten figure margins:
par(mar=c(0,0,4,0))

# plot soil colors
plot(gopheridge, name='hzname', color='soil_color')

# generate colors based on clay content
plot(gopheridge, name='hzname', color='clay')

# plot soils sorted by depth to contact
plot(gopheridge, name='hzname', plot.order=order(gopheridge$bedrckdepth))

# plot first 10 profiles
plot(gopheridge[1:10, ], name='hzname', color='soil_color')

# add rock fragment data to plot:
addVolumeFraction(gopheridge[1:10, ], colname='total_frags_pct')

# add diagnostic horizons
addDiagnosticBracket(gopheridge[1:10, ], kind='argillic horizon', col='red')

```

```
## End(Not run)
```

```
mapunit_geom_by_ll_bbox
```

Fetch Map Unit Geometry from SDA

Description

Fetch map unit geometry from the SDA website by WGS84 bounding box.

Usage

```
mapunit_geom_by_ll_bbox(bbox, source = 'sda')
```

Arguments

bbox	a bounding box in WGS coordinates
source	the source database, currently limited to soil data access (SDA)

Details

The SDA website can be found at <http://sdmdataaccess.nrcs.usda.gov>. See examples for bounding box formatting.

Value

A SpatialPolygonsDataFrame of map unit polygons, in WGS84 (long,lat) coordinates.

Note

It appears that SDA does not actually return the spatial intersecion of map unit polygons and bounding box. Rather, just those polygons that are completely within the bounding box / overlap with the bbox. This function requires the 'rgdal' package.

Author(s)

Dylan E Beaudette

References

<http://casoilresource.lawr.ucdavis.edu/>

Examples

```

# fetch map unit geometry from a bounding-box:
#
#           +----- (-120.41, 38.70)
#           |                   |
#           |                   |
# (-120.54, 38.61) -----+

## Not run:
# basic usage
b <- c(-120.54,38.61,-120.41,38.70)
x <- mapunit_geom_by_ll_bbox(b) # about 20 seconds

# note that the returned geometry is everything overlapping the bbox
# and not an intersection... why?
plot(x)
rect(b[1], b[2], b[3], b[4], border='red', lwd=2)

# get map unit data for matching map unit keys
in.statement <- format_SQL_in_statement(unique(x$MUKEY))
q <- paste("SELECT mukey, muname FROM mapunit WHERE mukey IN ", in.statement, sep="")
res <- SDA_query(q)

## End(Not run)

```

 parseWebReport

Parse contents of a web report, based on supplied arguments.

Description

Parse contents of a web report, based on supplied arguments.

Usage

```
parseWebReport(url, args, index = 1)
```

Arguments

url	Base URL to a LIMS/NASIS web report.
args	List of named arguments to send to report, see details.
index	Integer index specifying the table to return, or, NULL for a list of tables

Details

Report argument names can be inferred by inspection of the HTML source associated with any given web report.

Value

A data.frame object in the case of a single integer passed to index, a list object in the case of an integer vector or NULL passed to index.

Note

Most web reports are for internal use only.

Author(s)

D.E. Beaudette and S.M. Roecker

Examples

```
# pending
```

SDA_query

Soil Data Access Query

Description

Submit a query to the Soil Data Access (SDA) website in SQL, get the results as a dataframe.

Usage

```
SDA_query(q)
```

Arguments

q a valid T-SQL query surrounded by double quotes

Details

The SDA website can be found at <http://sdmdataaccess.nrcs.usda.gov> and query examples can be found at <http://sdmdataaccess.nrcs.usda.gov/QueryHelp.aspx>

Value

A dataframe containing the results. NULL is returned when queries result in 0 matches rows.

Note

This function requires the 'httr', 'jsonlite', and 'XML' packages

Author(s)

D.E. Beaudette

See Also

[mapunit_geom_by_ll_bbox](#)

Examples

```
# SSURGO export metadata:
## Not run:
q <- "SELECT areasymbol, saverest FROM sacatalog WHERE areasymbol LIKE 'CA%';"
x <- SDA_query(q)
x$saverest <- as.Date(x$saverest, format="%m/%d/%Y")
head(x)

## End(Not run)

# basic query:
## Not run:
res <- SDA_query("select cokey, compname, comppct_r
from component
where compname = 'yolo' and majcompflag = 'Yes' ")

## End(Not run)

# get component-level data for a specific soil survey area (Yolo county, CA)
## Not run:
q <- "SELECT
component.mukey, cokey, comppct_r, compname, taxclname,
taxorder, taxsuborder, taxgrtgroup, taxsubgrp
FROM legend
INNER JOIN mapunit ON mapunit.lkey = legend.lkey
LEFT OUTER JOIN component ON component.mukey = mapunit.mukey
WHERE legend.areasymbol = 'CA113'"

res <- SDA_query(q)

## End(Not run)

# get tabular data based on result from spatial query:
# requires raster and rgeos packages
## Not run:
library(raster) # suggested by soilDB
library(rgeos) # additional

# text -> bbox -> WKT
# xmin, xmax, ymin, ymax
b <- c(-120.9, -120.8, 37.7, 37.8)
p <- writeWKT(as(extent(b), 'SpatialPolygons'))
q <- paste0("SELECT mukey, cokey, compname, comppct_r
FROM component
WHERE mukey IN (
SELECT DISTINCT mukey
FROM SDA_Get_Mukey_from_intersection_with_WktWgs84('", p, "')")
```

```
    )  
    ORDER BY mukey, cokey, compct_r DESC")  
  
x <- SDA_query(q)  
  
## End(Not run)
```

SDA_query_features *Soil Data Access Spatial Query*

Description

Iterate over Spatial* object features and submit spatial queries to the SDA web-service.

Usage

```
SDA_query_features(x, id='pedon_id')
```

Arguments

x	a Spatial* object with more than 1 feature, any defined coordinate system
id	the column name in x that contains a unique ID for each feature

Details

The SDA website can be found at <http://sdmdataaccess.nrcs.usda.gov>. A detailed vignette can be found at https://r-forge.r-project.org/scm/viewvc.php/*checkout*/docs/soilDB/SDA-tutorial.html?root=aqp.

Value

A dataframe containing the results.

Note

This function requires the 'httr', 'jsonlite', 'XML', and 'rgeos' packages

Author(s)

D.E. Beaudette

seriesExtent	<i>Get/Display Soil Series Extent</i>
--------------	---------------------------------------

Description

Get or display the spatial extent of a named soil series using the Series Extent Explorer.

Usage

```
seriesExtent(s, timeout=60)
seriesExtentAsGmap(s, timeout=60, exp=1.25)
```

Arguments

s	the soil series name
timeout	time that we are willing to wait for a response, in seconds
exp	expansion factor used to expand Google Maps region

Details

Soil series extent data are downloaded from a static cache of GeoJSON files on SoilWeb servers. Cached data are typically updated annually.

Value

when calling seriesExtent, a SpatialPolygonsDataFrame object

Note

These function require the 'rgdal' and 'dismo' packages.

Author(s)

D.E. Beaudette

References

<http://casoilresource.lawr.ucdavis.edu/see>

Examples

```
## Not run:
# fetch series extent for the 'Amador' soil series
s <- seriesExtent('amador')
plot(s)

# fetch then plot the extent of the 'Amador' soil series
seriesExtentAsGmap('amador')
```

```
## End(Not run)
```

```
simplifyColorData      Simplify Color Data by ID
```

Description

Simplify multiple Munsell color observations associated with each horizon.

Usage

```
simplifyColorData(d, id.var = "phiid", ...)
mix_and_clean_colors(x, wt='pct', colorSpace='LAB')
```

Arguments

d	a data.frame object, typically returned from NASIS, see details
id.var	character vector with the name of the column containing an ID that is unique among all horizons in d
...	further arguments passed on to mix_and_clean_colors(), see details
x	a data.frame object containing sRGB coordinates associated with a group of colors to mix
wt	a character vector with the name of the column containing color weights for mixing
colorSpace	a character vector with the name of color space in which mixing is performed ("LAB" or "sRGB")

Details

This function is mainly intended for the processing of NASIS pedon/horizon data which may or may not contain multiple colors per horizon/moisture status combination. simplifyColorData will "mix" multiple colors associated with horizons in d, according to IDs specified by id.var, using "weights" (area percentages) specified by the wt argument to mix_and_clean_colors. Mixing is performed in the CIE LAB color space by default.

The simplifyColorData function can be applied to data sources other than NASIS by careful use of the id.var and wt arguments. However, d must contain Munsell colors split into columns named "colorhue", "colorvalue", and "colorchroma". In addition, the moisture state ("Dry" or "Moist") must be specified in a column named "colormoistst".

The mix_and_clean_colors function can be applied to arbitrary data sources as long as x contains sRGB coordinates in columns named "r", "g", and "b". This function should be applied to chunks of rows within which color mixtures make sense.

There are examples in [the KSSL data tutorial](#) and [the soil color mixing tutorial](#).

Author(s)

D.E. Beaudette

SoilWeb_spatial_query Get SSURGO Data via Spatial Query

Description

Get SSURGO Data via Spatial Query to SoilWeb

Usage

```
SoilWeb_spatial_query(bbox = NULL, coords = NULL, what = "mapunit", source = "soilweb")
```

Arguments

bbox	a bounding box in WGS84 geographic coordinates, see examples
coords	a coordinate pair in WGS84 geographic coordinates, see examples
what	data to query, currently ignored
source	the data source, currently ignored

Details

Data are currently available from SoilWeb. These data are a snapshot of the "official" data. The snapshot date is encoded in the "soilweb_last_update" column in the function return value. Planned updates to this function will include a switch to determine the data source: "official" data via USDA-NRCS servers, or a "snapshot" via SoilWeb.

Value

The data returned from this function will depend on the query style. See examples below.

Note

This function should be considered experimental; arguments, results, and side-effects could change at any time. SDA now supports spatial queries, consider using [SDA_query_features](#) instead.

Author(s)

D.E. Beaudette

Examples

```
# query by bbox
## Not run: SoilWeb_spatial_query(bbox=c(-122.05, 37, -122, 37.05))

# query by coordinate pair
## Not run: SoilWeb_spatial_query(coords=c(-121, 38))
```

unicode	<i>Convert coded values returned from NASIS and SDA queries to factors</i>
---------	--

Description

These functions convert the coded values returned from NASIS or SDA to factors (e.g. 1 = Alfisols) using the metadata tables from NASIS. For SDA the metadata is pulled from a static snapshot in the soilDB package (/data/metadata.rda).

Usage

```
unicode(df, invert = FALSE, db = "NASIS")
code(df, invert = TRUE, db = "NASIS")
```

Arguments

df	data.frame
invert	converts the code labels back to their coded values()
db	a string value specifying the soil database the data is coming from, which indicates whether or not to query metadata from NASIS or use soilDB-local snapshot

Details

These functions convert the coded values returned from NASIS into plaintext representation. The converted values from NASIS, or sourced from SDA, are upgraded to specifically-leveled factors using the metadata tables from NASIS. For SDA the metadata is pulled from a static snapshot in the soilDB package.

Value

A dataframe with the results.

Author(s)

Stephen Roecker

Examples

```
## Not run:
# query component by nationalmusym
comp = fetchSDA_component(WHERE = "nationalmusym = '2vzcp'")
s = site(comp$spc)
s = unicode(s, NASIS = FALSE)
levels(s$taxorder)

## End(Not run)
```

Index

*Topic **IO**

parseWebReport, 35

*Topic **datasets**

gSSURGO.chunk, 29

loafercreek, 31

*Topic **manip**

fetchHenry, 3

fetchNASIS, 7

fetchNASISLabData, 9

fetchOSD, 10

fetchPedinPC, 11

fetchSCAN, 13

fetchSDA_component, 14

get_colors_from_NASIS_db, 17

get_colors_from_pedin_db, 18

get_extended_data_from_pedin_db,
18

get_hz_data_from_NASIS_db, 19

get_hz_data_from_pedin_db, 20

get_lablayer_data_from_NASIS_db,
21

get_labpedon_data_from_NASIS_db,
21

get_phlabresults_data_from_NASIS_db,
22

get_site_data_from_NASIS_db, 23

get_site_data_from_pedin_db, 24

get_veg_from_AK_Site, 25

get_veg_from_MT_veg_db, 26

get_veg_from_NPS_PLOTS_db, 27

get_veg_other_from_MT_veg_db, 27

get_veg_species_from_MT_veg_db, 28

mapunit_geom_by_ll_bbox, 34

SDA_query, 36

SDA_query_features, 38

seriesExtent, 39

simplifyColorData, 40

SoilWeb_spatial_query, 41

unicode, 42

*Topic **package**

soilDB-package, 2

*Topic **utilities**

fetchKSSL, 5

fetchRaCA, 12

andic.subset (loafercreek), 31

cisne (loafercreek), 31

code (unicode), 42

fetchHenry, 3

fetchKSSL, 5

fetchLIMS_component

(fetchSDA_component), 14

fetchNASIS, 3, 7

fetchNASIS_component_data (fetchNASIS),
7

fetchNASIS_pedons (fetchNASIS), 7

fetchNASISLabData, 9

fetchOSD, 6, 10, 13

fetchPedinPC, 2, 3, 11

fetchRaCA, 12

fetchSCAN, 4, 13

fetchSDA_component, 14

format_SQL_in_statement (SDA_query), 36

get_chorizon_from_LIMS

(fetchSDA_component), 14

get_chorizon_from_SDA

(fetchSDA_component), 14

get_colors_from_NASIS_db, 17

get_colors_from_pedin_db, 18, 20

get_comonth_from_NASIS_db (fetchNASIS),
7

get_component_cogeomorph_data_from_NASIS_db
(fetchNASIS), 7

get_component_copm_data_from_NASIS_db
(fetchNASIS), 7

get_component_correlation_data_from_NASIS_db
(fetchNASIS), 7

- get_component_data_from_NASIS_db
(fetchNASIS), 7
- get_component_esd_data_from_NASIS_db
(fetchNASIS), 7
- get_component_from_LIMS
(fetchSDA_component), 14
- get_component_from_SDA
(fetchSDA_component), 14
- get_component_horizon_data_from_NASIS_db
(fetchNASIS), 7
- get_component_otherveg_data_from_NASIS_db
(fetchNASIS), 7
- get_copedon_from_NASIS_db (fetchNASIS),
7
- get_cosoilmoist_from_LIMS
(fetchSDA_component), 14
- get_cosoilmoist_from_NASIS
(fetchSDA_component), 14
- get_cosoilmoist_from_SDA
(fetchSDA_component), 14
- get_extended_data_from_NASIS_db
(fetchNASIS), 7
- get_extended_data_from_pedon_db, 18
- get_hz_data_from_NASIS_db, 17, 19, 20, 22,
23
- get_hz_data_from_pedon_db, 11, 18, 19, 20,
25, 26
- get_lablayer_data_from_NASIS_db, 21, 22
- get_labpedon_data_from_NASIS_db, 9, 21,
21
- get_mapunit_from_LIMS
(fetchSDA_component), 14
- get_mapunit_from_SDA
(fetchSDA_component), 14
- get_phlabresults_data_from_NASIS_db,
22
- get_phorizon_from_NASIS_db
(fetchNASIS), 7
- get_projectmapunit_from_NASIS
(fetchNASIS), 7
- get_site_data_from_NASIS_db, 17, 20, 22,
23
- get_site_data_from_pedon_db, 18–20, 24,
26
- get_text_notes_from_NASIS_db
(fetchNASIS), 7
- get_veg_from_AK_Site, 25, 25
- get_veg_from_MT_veg_db, 26, 28
- get_veg_from_NPS_PLOTS_db, 27
- get_veg_other_from_MT_veg_db, 26, 27, 28
- get_veg_species_from_MT_veg_db, 26, 28,
28
- get_vegplot_from_NASIS_db (fetchNASIS),
7
- get_vegplot_from_NASIS_db2
(fetchNASIS), 7
- getHzErrorsNASIS, 8, 9
- getHzErrorsNASIS (fetchNASIS), 7
- getHzErrorsPедonPC, 11
- getHzErrorsPедonPC (fetchPедonPC), 11
- gopheridge (loafercreek), 31
- gSSURGO.chunk, 29
- KSSL_VG_model, 29
- loafercreek, 3, 31
- mapunit_geom_by_ll_bbox, 34, 37
- metadata (unicode), 42
- mix_and_clean_colors
(simplifyColorData), 40
- parseWebReport, 35
- processSDA_WKT (SDA_query_features), 38
- SCAN_sensor_metadata (fetchSCAN), 13
- SDA_make_spatial_query
(SDA_query_features), 38
- SDA_make_spatial_query2
(SDA_query_features), 38
- SDA_query, 3, 36
- SDA_query_features, 38, 41
- seriesExtent, 39
- seriesExtentAsGmap (seriesExtent), 39
- simplifyFragmentData (fetchNASIS), 7
- simplifyColorData, 17, 40
- soilDB (soilDB-package), 2
- soilDB-package, 2
- soilDB.env (soilDB-package), 2
- SoilWeb_spatial_query, 41
- unicode, 42