Package 'sdam'

April 6, 2022

Type Package

Title Digital tools for the SDAM project at Aarhus University

Description
This package provides tools for performing analyses within Social Dynamics and complexity in the Ancient Mediterranean (SDAM), which is a research group based at Aarhus University.

Version 0.8.7

Date 2022-04-06

Depends R (>= 3.6.0)

Imports grImport2, multiplex

Suggests httr, rjson, grid, multigraph

Author Antonio Rivero Ostoic [aut, cre],
Adela Sobotkova [ctb],
Vojtech Kase [ctb],
Petra Hermankova [ctb]

Maintainer Antonio Rivero Ostoic <jaro@cas.au.dk>

R topics documented:

sdam-package *Digital tools for the SDAM project at Aarhus University*

Description

This package provides tools for performing analyses within Social Dynamics and complexity in the Ancient Mediterranean (SDAM), which is a research group based at Aarhus University.

Details

Package:	sdam
Type:	Package
Version:	0.8.7
Date:	6 April 2022
License:	CC BY-SA 4.0

Currently, it is possible with the "sdam" package to access data from the Epigraphic Database Heidelberg API with `get.edh()`, and the wrapper function `get.edhw()`. Most of the data is available in the `EDH` dataset attached to the package, which can be manipulated by using `edhw()` and `cln()`.

Besides, the `request()` function allows performing different types of HTTP requests to DEiC'S `https://sciencedata.dk` or another cloud repository with a customized URL address.

There is also the possibility to compute probabilities of existence with `prex()` with either the aoristic sum or count matching for observations for different periodization options. Function `plot.dates()` allows visualizing interval time events that can be manipulated by the internal function `dts` as illustrated in a vignette.

Another plotting function in "sdam" is found in `plot.map()` that allows visualizing maps of ancient Roman provinces that are found in `EDH`. This function uses datasets `rp`, `rpd`, `rpmp`, `rpmcd`, and `retn` for a Roman Empire transport network.

Similarity by simple matching among column vectors is achieved by the `simil()` function for making analyses of relations between assemblages and artifacts. Such similarity measure is the basis for function `edhwpd()` to organize the EDH dataset per province and dates.

Author(s)

Author: Antonio Rivero Ostoic [aut, cre], Adela Sobotkova [ctb], Vojtech Kase [ctb], Petra Hermankova [ctb]

Maintainer: Antonio Rivero Ostoic <jaro@cas.au.dk>

See Also

Vignettes

cln *Clean and re-encode a vector, list or dataframe*

Description

A function to re-encode Greek (and other) characters and to remove symbols at the end of a record.

Usage

```
cln(x, level, what, na.rm, case, repl)
```

Arguments

x	a vector, list or dataframe
level	optional clean level, either 0 for no-clean, default 1 or 2 (see details)
what	additional characters to clean (optional)
na.rm	Remove entries with NA data? (optional and logical)
case	case for text 1 for 1st uppercase, code2 for lowercase, code3 for uppercase (optional)
repl	data frame with text to replace (optional)

Details

This function is meant to re-encode Greek (and other) characters in the EDH set given either as list format, vector, or a dataframe produced with function edhw() for example. By default, symbols "?" "*" "+" placed at the end of each record are removed after the re-encoding. However, when level is 0 only re-encoding is performed, and level 2 is either to force an extra iteration in the re-encoding, removing extra spaces, or clean what is invoked. With repl, is possible to replace a list of text in two columns, for 'text to replace' and for 'text that replace'

Value

Depending on the input, a vector, list or dataframe.

Author(s)

Antonio Rivero Ostoic

See Also

edhw, get.edh

Examples

```
# clean Greek characters
cln("Caesar?*+")
```

EDH *Epigraphic Database Heidelberg Dataset*

Description

This is a data set retrieved from the Epigraphic Database Heidelberg API repository.

Usage

```
data("EDH")
```

Format

A list object of 84701 records (until 10-11-2020) with at least one of the following 47 (or more) names in the EDH list:

"ID", "commentary", "fotos", "country", "depth", "diplomatic_text", "edh_geography_uri", "findspot", "findspot_ancient", "findspot_modern", "geography", "height", "id", "language", "last_update", "letter_size", "literature", "material", "military", "modern_region", "not_after", "not_before", "people" (which is a list with: "person_id", "nomen", "cognomen", "praenomen", "name", "gender", "status", "tribus", "origo", "occupation", "age: years", "age: months", "age: days"), "present_location", "province_label", "religion", "responsible_individual", "social_economic_legal_history", "transcription", "trismegistos_uri", "type_of_inscription", "type_of_monument", "uri", "width", "work_status", and "year_of_find".

Source

https://edh-www.adw.uni-heidelberg.de/data/api

See Also

get.edh, get.edhw,edhw

edhw *Wrapper function for manipulation of* EDH *dataset*

Description

A function to obtain variable data and perform transformations on the Epigraphic Database Heidelberg EDH dataset.

Usage

```
edhw(x = "EDH", vars, as = c("df", "list"), type = c("long", "wide", "narrow"),
     split, select, addID, limit, id, na.rm, ldf, province, gender, rp, ...)
```

Arguments

x	A list object name with fragments of the EDH dataset (optional)
vars	Variables of interest from x; if x=NULL, the entire EDH dataset is taken (optional, vector)
as	Format to return the output. Currently either as a "list" or a data frame "df" object.
type	Type format of data frame. Currently either "long" or "wide" ("narrow" not yet implemented)
split	Divide the data into groups by id? (optional and logical)
select	Choose "people" variables (optional, vector)
addID	Add identification to the output? (optional and logical)
limit	Limit the returned output. Ignored if id is specified (optional, integer or vector)
id	Select only hd_nr records (optional, integer or character)
na.rm	Remove entries with NA data? (optional and logical)
ldf	is x list of data frames? (optional and logical)
province	Roman province in EDH as in rp dataset
gender	people gender in EDH (male or female)
rp	customized list of Roman provinces as in rp dataset
...	optional arguments if needed.

Details

This is an interface to extract attribute variables in vars from the EDH dataset attached to this package. However, the input in x can be fragments of the EDH dataset or from the Epigraphic Database Heidelberg API obtained by functions get.edh() or get.edhw() with the "rjson" format, or transformed data organized, for example, by provinces. When x is explicit, it must be at least a list object with a comparable structure to the EDH dataset.

Through vars argument and return the output either as a list with list or a data frame with df. In case argument vars is missing, then all entries in x are taken.

By default, a list object is returned, with or without an 'ID' identification provided by the addID argument. When the input list is converted into a data frame, the ordering of the variables is given alphabetically. If desired, it is also possible to remove missing data from the output by activating na.rm and work with complete cases.

Arguments id and limit serve to reduce the returned output either to some Epigraphic Database number or numbers, which are specified by hd_nr, or else by limiting the amount of the returned output. limit here is like the limit argument of function get.edh(), but in this case the offset can be specified as a sequence. While limit is a faster way to get to entries in the EDH dataset, argument id is for referring to precisely one or more hd_nrs in the Epigraphic Database Heidelberg API.

Component "people" is a separated list in the EDH dataset, and it should be considered as a separate case from the rest of the variables. In the case that the output is a data frame, the default output is a 'long' type table; that is records can appear in different rows and each variable is assigned into a single column, and with this option is possible to select "people" variables like gender and origin.

By setting "wide" in type, it is possible to place the different people from a single entry column by column in the data frame and each record has a single row. Finally, argument split allows dividing the data in the data frame into groups by 'id', which corresponds to the HD number of inscription in the EDH dataset.

Ad hoc arguments are the EDH entries province and gender for entering a Roman province and people's gender in x as a data frame; otherwise these arguments are ignored. When province is used, it is possible to refer to a customized list of provinces with argument rp; otherwise dataset "rp" is employed.

Argument ldf is a flag when the input is a created list of data frames that is organized by variables rather than by records as in EDH.

Value

A list or a data frame with a long or wide format, depending on the input arguments.

Note

When choosing "people" variables with select with a data frame output, then this attribute must be in vars.

Author(s)

Antonio Rivero Ostoic

References

https://edh.ub.uni-heidelberg.de/data/api

See Also

get.edh, get.edhw, edhwpd, prex, plot.dates, EDH

Examples

```
## Not run:
## load dataset
data(EDH)

## make a list for three variables in 'EDH' for first 4 entries
edhw(vars=c("type_of_inscription", "not_after", "not_before"), limit=4 )

## as before, but also select 'gender' from 'people'
edhw(vars=c("people", "not_after", "not_before"), select="gender", limit=4 )
## End(Not run)
```

---

edhwpd *Organize EDH dataset province and dates by similarity*

Description

Wrapper function to organize EDH dataset province and dates by simple match similarity.

Usage

```
edhwpd(x = NULL, vars, province, dates, clean)
```

Arguments

x	typically fragments of EDH dataset or database API (optional, list)
vars	variables or attributes to be chosen from x (vector)
province	choosen EDH province abbreviation
dates	vector with TAQ and TPQ (optional)
clean	whether to remove special characters in text (optional and logical)

Details

As with function edhw, this is an interface to extract attribute variables in vars from the EDH or similar dataset if x is not specified. The aim is to organize data per province and dates by simple match similarity among records.

The Roman Empire province is the abbreviation used in the value given by function get.edh and which is in "rp" dataset.

Argument dates is optional to specify the variables for time intervals (TAQ and TPQ) that in EDH are "not_after" and "not_before".

Argument clean applies function cln to the province data frame with the chosen variables to remove special characters such as ?*+ and, if needed, re-encode the text.

The output is a list of data frames with similar arguments by descending matches. The records with one or less similarity matches (or having NA attribute values) are placed in the last data frame of the list.

Value

List of data frames of a EDH class object that includes the province with the number of records.

Author(s)

Antonio Rivero Ostoic

See Also

edhw, get.edh, cln

Examples

```
## Not run:
## load dataset
data(EDH)

## extract province & dates with a single variable attribute from EDH
edhwpd(vars="type_of_inscription", province="Rom", dates=c("not_after", "not_before"))

## End(Not run)
```

get.edh *Get data from the Epigraphic Database Heidelberg API*

Description

A function to obtain data from the Epigraphic Database Heidelberg API repository.

Usage

```
get.edh(search = c("inscriptions", "geography"), url =
                  "https://edh.ub.uni-heidelberg.de/data/api", hd_nr,
                  province, country, findspot_modern, findspot_ancient,
                  year_not_before, year_not_after, tm_nr, transcription,
                  type, bbox, findspot, pleiades_id, geonames_id,
                  offset, limit, maxlimit = 4000, addID, printQ)
```

Arguments

search	Whether the search is on inscriptions *or* on geography.
url	Open data repository API
hd_nr	HD number of inscription
province	Ancient Roman province name
country	Actual country name
findspot_modern	Actual location name findspot
findspot_ancient	Ancient location name findspot
year_not_before	Year, not before (integer, BC years are negative)
year_not_after	Year, not after (integer, BC years are negative)
tm_nr	Trismegistos' database number (?)
transcription	Automatic leading and trailing truncation (brackets are ignored)
type	Type of inscription (case insensitive)
bbox	Bounding box with character format bbox = "minLong,minLat,maxLong,maxLat"
findspot	Level of village, street etc. (add leading and/or trailing)
pleiades_id	Pleiades identifier of a place (integer)
geonames_id	Geonames identifier of a place (integer)
offset	Clause to specify which row to start from retrieving data (optional and integer)
limit	Clause to limit the number of results (optional and integer)
maxlimit	Maximum limit of the query (integer, default 4000)
addID	Add identification to the output? (optional and logical)
printQ	Also print query? (optional and logical)

Details

This function is for the new [EDH] database [API] in year 2022.

Since with the `inscriptions` option the `id` "component" of the output list is not with a numeric format, then the function adds an `ID` at the beginning of the list with the identifier with a numerical format.

Notice that `hd_nr` is not the same as `ID` nor `id`.

Use function `get.edhw` in case you want to grab several items.

A list with the of valid values from the EDH API for the ancient Roman provinces are

"Ach"	Achaia	"Cor"	Corsica	"Mes"	Mesopotamia
"Aeg"	Aegyptus	"Cre"	Creta	"MoI"	Moesia inferior
"Aem"	Aemilia (Regio VIII)	"Cyp"	Cyprus	"MoS"	Moesia superior
"Afr"	Africa Proconsularis	"Cyr"	Cyrene	"Nar"	Narbonensis
"AlC"	Alpes Cottiae	"Dac"	Dacia	"Nor"	Noricum
"AlG"	Alpes Graiae	"Dal"	Dalmatia	"Num"	Numidia
"AlM"	Alpes Maritimae	"Epi"	Epirus	"PaI"	Pannonia inferior
"AlP"	Alpes Poeninae	"Etr"	Etruria (Regio VII)	"PaS"	Pannonia superior
"ApC"	Apulia et Calabria (Regio II)	"Gal"	Galatia	"Pic"	Picenum (Regio V)
"Aqu"	Aquitania	"GeI"	Germania inferior	"Rae"	Raetia
"Ara"	Arabia	"GeS"	Germania superior	"ReB"	Regnum Bospori
"Arm"	Armenia	"HiC"	Hispania citerior	"Rom"	Roma
"Asi"	Asia	"Inc"	Provincia incerta	"Sam"	Samnium (Regio IV)
"Ass"	Assyria	"Iud"	Iudaea	"Sar"	Sardinia
"Bae"	Baetica	"LaC"	Latium et Campania (Regio I)	"Sic"	Sicilia, Melita
"Bar"	Barbaricum	"Lig"	Liguria (Regio IX)	"Syr"	Syria
"Bel"	Belgica	"Lug"	Lugdunensis	"Thr"	Thracia
"BiP"	Bithynia et Pontus	"Lus"	Lusitania	"Tra"	Transpadana (Regio XI)
"BrL"	Bruttium et Lucania (Regio III)	"LyP"	Lycia et Pamphylia	"Tri"	Tripolitania
"Bri"	Britannia	"MaC"	Mauretania Caesariensis	"Umb"	Umbria (Regio VI)
"Cap"	Cappadocia	"MaT"	Mauretania Tingitana	"Val"	Valeria
"Cil"	Cilicia	"Mak"	Macedonia	"VeH"	Venetia et Histria (Regio X

And the valid values for `country` entries are abbreviated country names where the inscription was located.

"ad"	Andorra	"gr"	Greece	"pl"	Poland
"al"	Albania	"hr"	Croatia	"pt"	Portugal
"am"	Armenia	"hu"	Hungary	"rks"	Kosovo
"at"	Austria	"il"	Israel	"ro"	Romania
"az"	Azerbaijan	"iq"	Iraq	"rs"	Serbia
"ba"	Bosnia and Herzegovina	"it"	Italy	"ru"	Russia
"be"	Belgium	"jo"	Jordan	"sa"	Saudi Arabia
"bg"	Bulgaria	"kg"	Kyrgyzstan	"sd"	Sudan
"ch"	Switzerland	"kz"	Kazakhstan	"se"	Sweden
"cy"	Cyprus	"lb"	Lebanon	"si"	Slovenia
"cz"	Czech Republic	"li"	Liechtenstein	"sk"	Slovakia
"de"	Germany	"lu"	Luxembourg	"sm"	San Marino
"dk"	Denmark	"ly"	Libyan Arab Jamahiriya	"sy"	Syrian Arab Republic
"dz"	Algeria	"ma"	Morocco	"tj"	Tajikistan
"eg"	Egypt	"mc"	Monaco	"tn"	Tunisia

"es"	Spain	"md"	Moldova	"tr"	Turkey
"fr"	France	"me"	Montenegro	"ua"	Ukraine
"gb"	United Kingdom	"mk"	Macedonia	"uz"	Uzbekistan
"ge"	Georgia	"mt"	Malta	"va"	Vatican City State
"gi"	Gibraltar	"nl"	Netherlands	"ye"	Yemen

Value

A list object with at least one the following items:

"ID" (Optional), only if addID is set to TRUE.

"commentary"

"fotos"

"country"

"depth"

"diplomatic_text"

"edh_geography_uri"

"findspot"

"findspot_ancient"

"findspot_modern"

"geography"

"height"

"id"

"language"

"last_update"

"letter_size"

"literature"

"material"

"military"

"modern_region"

"not_after"

"not_before"

"people" This item is another list with at least one the following items:

"person_id"

"nomen"

"cognomen"

"praenomen"

```
                    "name"
                    "gender"
                    "status"
                    "tribus"
                    "origo"
                    "occupation"
                    "age: years"
                    "age: months"
                    "age: days"

  "present_location"

  "religion"
  "province_label"

  "responsible_individual"

  "social_economic_legal_history"

  "transcription"

  "trismegistos_uri"

  "type_of_inscription"

  "type_of_monument"

  "uri"
  "width"
  "work_status"
  "year_of_find"
```

And also the query is printed if specified by `printQ`.

Warning

Queries with more than 4000 records can produce a timeout from the server, and a *Warning* message is produced.

Author(s)

Antonio Rivero Ostoic

References

https://edh.ub.uni-heidelberg.de/data/api

See Also

`get.edhw`, `edhw`, `edhwpd`, `plot.map`, `simil`

Examples

```
## Not run:
# get inscriptions from EDH API data
get.edh(findspot_modern="madrid")
## End(Not run)
```

`get.edhw` *Wrapper to get data from the Epigraphic Database Heidelberg API*

Description

A wrapper function to obtain data from the Epigraphic Database Heidelberg repository.

Usage

```
get.edhw(file = NULL, hd_nr, ...)
```

Arguments

`file`	JSON file with EDH data (optional)
`hd_nr`	HD number of inscriptions
`...`	Additional arguments

Details

This is a wrapper function to obtain a sample data from the Epigraphic Database Heidelberg API repository by their HD numbers or, alternatively, a file with a valid format JSON can be specified in `file`.

In any case, the JSON output will be converted into a list object with the `rjson` package.

Value

A list of lists object with the items described in `get.edh`.

Note

Large samples can take a lot of time.

Author(s)

Antonio Rivero Ostoic

References

https://edh.ub.uni-heidelberg.de/data/api

See Also

`get.edh`, `simil`

Examples

```
## get 10 records from EDH API data
## Not run:
get.edhw(hd_nr=1:10)
## End(**Not run**)
```

`plot.dates` *Plot interval dates*

Description

A function to plot interval dates.

Usage

```
plot.dates(x, y, file = NULL, taq, tpq, id, out, main = NULL,
           xlab = NULL, ylab = NULL, xlim = NULL, axes = TRUE,
           cex, pch, col, lwd, lty, alpha, ...)
```

Arguments

`x`	dataset as a data frame object of variables and observations.
`y`	optional identifiers (vector).
`file`	path to file for a PDF format (optional)
`taq`	*terminus ante quem*
`tpq`	*terminus post quem*
`id`	IDs as variable or rownames in dataset `x`
`out`	number of outliers to omit (integer or vector where first entry id for latest date)
`main`	plot's main tile
`xlab`	plot's `x` label
`ylab`	plot's `y` label
`xlim`	plot's `x` limits
`axes`	plot's axes (logical)
`cex`	size of `pch`
`pch`	symbol for `taq` and `tpq`
`col`	color of `pch`
`lwd`	width of time interval segments
`lty`	shape of time interval segments
`alpha`	alpha transparency for time interval segments
`...`	additional optional parameters

Details

This plot function is for time interval segments given in the dataset `x`, which is given as a data frame or as a "tibble" class object.

Value

A graphical plot.

Note

If x is NULL, then EDH dataset is taken.

Author(s)

Antonio Rivero Ostoic

See Also

get.edh, edhw, prex.

Examples

```
#TBD.
```

---

plot.map *Plot cartographical map*

Description

A function to plot a cartographical map, initially of a Roman province or an Italian region.

Usage

```
plot.map(x = NULL, type = c("plain", "rp", "si", "tetra", "med"), settl, roads, shipr,
         main, cap, date, name, fsize, fcol, fsize2, fcol2, xd, yd, new, ...)
```

Arguments

x	(character) acronym of ancient Roman province or Italian region (see get.edh)
type	Type of cartographical map: plain rp Roman provinces si Senatorial-Imperial tetra Tetrarchy med Mediterranean sea
settl	(logical for cartographical map) Display settlements?
roads	(logical for cartographical map) Display roads?
shipr	(logical for cartographical map) Display shipping routes?
main	(optional for cartographical map) Plot's main title
cap	(logical for provinces) display caption?
date	(logical for provinces) display date?
name	(logical for provinces) display province title name?
fsize	(optional) font size in main title

fcol	(optional) font color in main title
fsize2	(optional) font size in date
fcol2	(optional) font color in date
xd	(optional) *x* positioning for date
yd	(optional) *y* positioning for date
new	(optional) whether or not plotted map has superimposed graphics
...	additional optional parameters

Details

This plot function is for creating cartographical maps of ancient provinces and Italian regions of the Roman Empire around year 117 AD. The input data x can be a character vector, but this is intended for a recording output. By default, arguments name and cap are set to TRUE while date is set to FALSE.

The type argument allows plotting cartographical maps related to the Roman Empire and Mediterranean sea.

Value

A cartographical map plot with a name, and a caption with an approximate province establishment date.

Note

Positions for caption and date are fixed for a PDF output.

Author(s)

Antonio Rivero Ostoic

See Also

rpmp, rpmcd, get.edh, rp, EDH

Examples

```
#TBD.
```

---

prex *Compute probability of existence*

Description

A function to compute the probability of existence of events.

Usage

```
prex(x, taq, tpq, vars, bins = NULL, cp, aoristic = TRUE,
     weight = 1, DF, out, plot = FALSE, main = NULL, ylim, ...)
```

Arguments

x	list or data frame object of variables and observations.
taq	*terminus ante quem* (TAQ)
tpq	*terminus post quem* (TPQ)
vars	boundaries of existence in x (vector for TAQ and TPQ)
bins	length of the break (integer)
cp	chronological phase
aoristic	return aoristic sum? (logical)
weight	weight to observations
DF	return also data frame with observations? Ignored for plot (logical and optional)
out	number of outliers to omit (integer or vector where first entry id for latest date)
plot	plot the results? (logical and optional)
main	plot's main title (optional)
ylim	limit in y-axis (optional, only with plot)
...	additional optional parameters

Details

Currently, the probability of existence is the *aoristic sum* computed across events for portions of time delimited by a TAQ in taq and TPQ in tpq, or else by the boundaries of existence in vars.

In case that bins is NULL, then the time breaks take the chronological periods in cp, which by default is "bin5" or five-periods for the EDH dataset. Another built-in option is "bin8" for eight chronological periods, but cp is open for other periodization models as long as they are recorded as a list object.

When aoristic is set to FALSE, then a simple matching of only TAQ and TPQ is computed from x.

Value

A data frame with values according to either bins or cp.

Author(s)

Antonio Rivero Ostoic

References

For aoristic sum: Crema, E. (2012) "Modelling temporal uncertainty in archaeological analysis" *J Archaeol Method Theory*.

For default chronological periods: see *Bevan et al*, 2013 (doi: 10.1111/j.1475-4754.2012.00674.x)

See Also

edhw, plot.dates

Examples

```
#TBD.
```

request *Perform an HTTP request*

Description

A function to perform an HTTP request

Usage

```
request(file, URL = "https://sciencedata.dk", method = c("GET", "POST", "PUT", "DELETE"),
      anonymous = FALSE, cred = NULL, path = "/files", subdomain = NULL, force = FALSE,
         rm.file, ...)
```

Arguments

file	the request file
URL	protocol and domain of the url
method	the http verb for the object
anonymous	unauthenticated user? (logical)
cred	authentication credentials (vector with username and password)
path	path to add to the url (optional)
subdomain	subdomain to add to the url (optional)
force	force remote file overwriting? (optional)
rm.file	remove file in local machine? (optional and logical)
...	extra parameters if required

Details

request is basically a HTTP request, first aimed to interact with DEiC's (Danish e-Infrastructure Cooperation) https://sciencedata.dk. However, it is possible to specify the URL path and subdomain if necessary.

There are two types of folders in DEiC's https://sciencedata.dk that are *personal* and *shared* folders and both requires authentication with credentials.

The *path* to the shared folders where the files are located must be specified with the path argument. However, for personal folders is the file argument that includes the path information. Many times, DEiC's https://sciencedata.dk places the data on a *subdomain*, and for some methods like PUT it is required to specify the subdomain as well.

When a file already exists on the remote server, there is a prompt question for overwriting the file when the PUT method is invoked, and by activating argument force we can prevent confirmation and replace the file.

In case that accessing the server requires basic authentication, then package "tcltk" may be needed as well to input the credentials with a widget prompt, and there is the cred argument for performing a basic authentication without a prompt. Public folders can be accessed as anonymous user.

Value

Depends on the method, an action on the server site. A *Response* message is returned when the method is PUT with the url and items Date, Status, Content-Type. Currently, method POST is not supported at *sciencedata.dk*.

Note

Aliases for this function are `sddk()` and `SDDK()`.

Author(s)

Antonio Rivero Ostoic

See Also

https://sciencedata.dk

https://mplex.github.io/cedhar/Sciencedata_dk.html

Examples

```
## get a public file from remote server as anonymous user
## Not run:
request("filename.extension", method="GET", anonymous=TRUE)
## End(Not run)

## put a file in remote server
## Not run:
request("filename.extension", method="PUT")
## End(Not run)

## put an existing file in remote server and force overwriting
## Not run:
request("filename.extension", method="PUT", force=TRUE)
## End(Not run)

## put an existing file in remote server and remove file from local machine
## Not run:
request("filename.extension", method="PUT", rm.file=TRUE)
## End(Not run)


## remove a file in remote server
## Not run:
request("filename.extension", method="DELETE")
## End(Not run)
```

---

retn *Roman Empire transport network and Mediterranean sea*

Description

This is a list of lists with specifications to plot different cartographical maps of the Roman Empire and the Mediterranean sea with transport network including settlements, roads, and shipping routes.

Usage

```
data("retn")
```

Format

A list of lists object with the shape data in different slots for 4 cartographical maps of the Roman Empire with names `rcoast` for a plain map, `rpcoast` for a map with provinces, `rpsi` for a map with senatorial and imperial provinces, and `rptetra` for a tetrarchy map. Three components in `retn` have coordinates for settlements `nds`, roads `rds`, and shipping routes `srs` for this maps.

In addition, the dataset has a cartographical map of the Mediterranean sea in `med` where settlements and transport network is yet to complete.

Source

DARMC, Center for Geographic Analysis, Harvard University

Rodrigue, Comtois, Slack. (2013) *The geography of transport systems*. Routledge

https://commons.wikimedia.org/wiki/File:RomanEmpire_117.svg

https://commons.wikimedia.org/wiki/File:Roman_provinces_trajan.svg

https://commons.wikimedia.org/wiki/File:Regioni_dell'Italia_Augustea.svg

See Also

`plot.map`, `rp`, `rpmp`, `rpmcd`

---

rp *Roman province names and acronyms as in EDH*

Description

This is a list with Roman province names and acronyms as in the Epigraphic Database Heidelberg, EDH.

Usage

```
data("rp")
```

Format

A list object of 66 Roman provinces named with `"province_label"`.

Source

https://edh-www.adw.uni-heidelberg.de/data/api/terms/province

See Also

`get.edh`, `EDH`, `edhw`, `retn`, `rpmp`,

rpd *Roman provinces dates from EDH*

Description

This is a list with Roman province dates from the Epigraphic Database Heidelberg, EDH.

Usage

```
data("rpd")
```

Format

A list object of 66 Roman provinces with dates. Each list component has a vector for the province containing of the dates: the average, min TAQ, max TPQ, and the average length time span. Components in the list have also an attribute class with the `"HD_nr"` entries of the province.

Source

https://edh-www.adw.uni-heidelberg.de/data/api/

See Also

`rmids`, `rp`, `EDH`, `edhwpd`, `get.edh`

rpmcd *Caption maps and affiliation dates of Roman provinces*

Description

This is a list with specifications to plot caption maps of 59 Roman provinces (year 117 AD) and Italian regions under emperor Augustus (year 27 BC).

Usage

```
data("rpmcd")
```

Format

`rpmcd` is a list of lists for each province or region with two main components. One component is a list with shape data for a cartographical map caption in different slots, and the second component has an affiliation date for each Roman province or when the territory became Roman province. `names(rpmcd)` has the acronyms according to `EDH` dataset.

Source

https://commons.wikimedia.org/wiki/File:RomanEmpire_117.svg

https://commons.wikimedia.org/wiki/File:Roman_provinces_trajan.svg

https://commons.wikimedia.org/wiki/File:Regioni_dell'Italia_Augustea.svg

See Also

retn, rpmp, plot.map, rp, EDH

---

rpmp *Maps of ancient Roman provinces and Italian regions*

Description

This is a list with specifications to plot cartographical maps of ancient Roman provinces and Italian regions.

Usage

```
data("rpmp")
```

Format

A list of lists object of 59 Roman provinces and Italian regions in year 117AD, and where `names(rpmp)` gives the province acronyms according to EDH dataset. Each province in rpmp has a two-length list with the province name and the shape data for a cartographical map in different slots.

Source

https://commons.wikimedia.org/wiki/File:RomanEmpire_117.svg

https://commons.wikimedia.org/wiki/File:Roman_provinces_trajan.svg

https://commons.wikimedia.org/wiki/File:Regioni_dell'Italia_Augustea.svg

See Also

plot.map, rpmcd, rp, retn, EDH

---

simil *Similarity between (column) vectors*

Description

A function to compute similarity between vectors, which can arise from columns in a data frame or list entries.

Usage

```
simil(x, vars, uniq, diag.incl)
```

Arguments

x	A list or a data frame object
vars	Column(s) in x representing variable attributes (vector)
uniq	unique entries? (optional and logical)
diag.incl	include entries in matrix diagonal? (optional and logical)

Details

At this point, the ID column in the input represents the labels of the nodes. In case that an ID column does not exists, then the first column is taken provided that there are not duplicated entry names.

Value

A valued matrix with similarities among units by simple matching.

Note

Other similarity measures will be added in the near future.

Author(s)

Antonio Rivero Ostoic

See Also

`get.edh`

Examples

```
# TBD
```

Index