Package ‘graphing’

December 31, 2021

Title graphing

Version 0.0.1

Description
Collection of descriptive graphing functions and a function for ORCiD to name conversion.

Depends R (>= 3.5.0)

Imports utils, stats, grDevices, tm, igraph, countrycode, rworldmap, geosphere, rgeos, RColorBrewer, rorcid, future.apply

License CC0

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

R topics documented:

ani.barplot *Animated Barplot*

Description

Creates an animation of a one dimensional frequency table or vector. Animation can be saved to a folder in several formats.

Usage

```
ani.barplot(
  x,
  col = "lightblue",
  ylab = "",
  xlab = "",
  main = "",
  ylim = c(0, max(x) + max(x)/6),
  box = TRUE,
  names = "",
  cex.names = par("cex.axis"),
  border = "black",
  space = 0.4,
  text = TRUE,
  cex = 1,
  coltext = 1,
  pre = "n =",
  post = "%",
  format = "png",
  digit = 2,
  frames = 100,
  folder = "",
  ani.text = TRUE,
  width = 1920,
  height = 1080,
  point = 8,
  res = 300,
  bg = TRUE,
  bg.from = "grey",
  bg.to = "white",
  bg.increase = TRUE,
  axes = TRUE,
  cex.axis = par("cex.axis")
)
```

Arguments

x	a vector or one dimensional frequwncy table of values describing the bars which make up the plot.

`col` a vector of colors for the bars. By default col is set to "lightblue".

`ylab` a label for the y axis

`xlab` a label for the x axis

`main` title of plot

`ylim` limits for the y axis.

`box` a logical value. If FALSE no box around plot will be drawn

`names` a vector of names to be plotted below each bar. If this argument is left empty, the names are taken from the names freuqency table of x

`cex.names` expansation of names below bars

`border` color of bar border

`space` space between bars

`text` a logical value. If TRUE text will be drawn on bars

`cex` expansation of text on bars

`coltext` text color

`pre` pre text of absolute frequencies on bars

`post` post text of absolute frequencies on bars

`format` the image output format e.g.: "png", "pdf", "jpeg", etc.

`digit` the numer of decimal places to display relative frequencies with e.g.: digits=3 -> 22,2

`frames` animation length in frames

`folder` folder to save animation as png frames to. If "" no images will be saved

`ani.text` a logical value. If TRUE text on bars will be animated as well. If FALSE no text on bars will apear.

`width` png width (only works if format="png")

`height` png height (only works if format="png")

`point` png point size (only works if format="png")

`res` png resolution px/in (only works if format="png")

`bg` a logical value. If TRUE background will be drawn.

`bg.from` background color starting color

`bg.to` background color end color

`bg.increase` a logical value. If TRUE background color will increase from bg.from to bg.to

`axes` a logical value. If TRUE axes will be drawn.

`cex.axis` text size of axis

Examples

```
x<-sample(paste("group",letters[1:5]),200,T)
ani.barplot(prop.table(table(x)))
```

bp3d *A barplot for 3 dimensional contingency tables*

Description

draws the conditioned freuqencies of a categorial variable on two further categorial variables

Usage

```
bp3d(
  x,
  z,
  y,
  main = "",
  ylab = "frequencies",
  xlab = "",
  col = heat.colors(length(levels(y)) + 2)[1:length(levels(y))],
  space = 0.1,
  factor_space = 3,
  left_space = 0,
  box = T,
  boxcol = c("white", "grey"),
  legend = TRUE,
  locator = "topright"
)
```

Arguments

`x`	main categorical variable on x-axis
`z`	categorical sub variable on x axis
`y`	dependend categorical variable
`main`	title
`ylab`	y axis label
`xlab`	x axis label
`col`	colors of bar segments
`space`	space between bars within groups
`factor_space`	factor of space between groups
`left_space`	left space of bars
`box`	draw boxes behind goups of x
`boxcol`	color of boxes behind groups
`legend`	draw legend
`locator`	location of legend: "topright", "right" "bottomright"

Examples

```
x <- sample(paste(”time”,1:4),240,T)
z <- rep(c(”A”,”B”),each=120)
y <- paste(”answer”,sample(1:7,240,T))
bp3d(x,z,y,main=”bp3d()”,xlab=”treatment group”)
```

`catplot` *catplot*

Description

Draws conditioned or unconditioned frequency table of two multi categorial variables

Usage

```
catplot(
  x,
  y,
  margin = 1,
  main = "",
  xlab = "",
  ylab = "",
  cex.lab = 1,
  marginleft = 5,
  cexmax = 4
)
```

Arguments

`x`	multi categorial variable
`y`	multi categorial variable
`margin=1`	conditioned frquencies (1: cond. on y, 2: cond. on x, NULL: unconditioned)
`main=""`	main title
`xlab=""`	x axis label
`ylab=""`	y axis label
`marginleft=5`	left margin space
`cexmax=2`	max size of squares

Examples

```
x<-sample(2010:2020,1000,T,.2+(1:11)*.5)
y<-sample(paste(”group”,LETTERS[1:10]),1000,T)
catplot(x,y,main=”catplot()”,xlab=”year”)
```

CIV *Confidence Intervall Violin Plot*

Description

draws observations in metric variable for multiple groups, the group means, the 1-alpha confidence intervalls of the means and a density estimator (violin) of the distribution for every group.

Usage

```
CIV(
  x,
  y,
  alpha = 0.05,
 ylim = range(y) + c(-(range(y, na.rm = T)[2] - range(y, na.rm = T)[1])/5, (range(y,
    na.rm = T)[2] - range(y, na.rm = T)[1])/5),
  ylab = "",
 xlab = paste("|---| ", 100 * (1 - alpha), "% - confidence intervall", sep = ""),
  main = "",
  pch = 1 + as.numeric(factor(x)),
  col = "blue",
  colpoints = "grey",
  cex = 0.7,
  names = levels(factor(x)),
  jit = 0.2
)
```

Arguments

x	a categorial variable (independend variable)
y	a numeric variable (dependend variable)
alpha	error probability of confidence intervall
ylim	y axis limits
ylab	y axis label
xlab	x axis label
main	title
pch	point character
col	mean confidence intervall color
colpoints	point color
cex	point size
names	names of groups on x axis
jit	jitter intensity

Examples

```
x<-sample(paste("group",1:4),100,T)
y<-10+as.numeric(factor(x))*2+rnorm(length(x))
CIV(x,y,main="Confidence Intervall Violin Plot")
```

igraph2 *igraph2*

Description

draw network graph from list input

Usage

```
igraph2(
  x,
  freq = FALSE,
  label.cex = 1.7,
  vertex.size = 5,
  split2words = FALSE,
  rm.punctuation = FALSE,
  lowerize = FALSE,
  stemming = FALSE,
  capitalize = FALSE,
  rm.stopwords = FALSE,
  rm.numbers = FALSE,
  tkplot = FALSE,
  seed = NULL,
  min.n = 1,
  min.freq = 0,
  max.freq = 1,
  layout = 1
)
```

Arguments

`x`	a list with connected elements as vector
`freq`	logical. If TRUE displays frequency of connections in edge labels
`label.cex`	numerical. Node label size
`vertex.size`	numerical. Size of verices
`split2words`	logical. If TRUE each element is further split at spaces
`rm.punctuation`	logical. If TRUE punctuation/spaces are removed
`lowerize`	logical. If TRUE lowerize to unify elements
`stemming`	logical. If TRUE enables stemming

`capitalize`	logical. If TRUE capitalize to display
`rm.stopwords`	logical. If TRUE uses stop word removal
`rm.numbers`	logical. IF TRUE excludes numbers
`tkplot`	logical. If TRUE enables interactive adjustment with tkplot
`seed`	numerical. Set seed for reproducible graphs
`min.n`	numerical. Absolute minimum of involvement per element
`min.freq`	numerical. Relative minimum of involvement per element
`max.freq`	numerical. Relative maximum of involvement per element
`layout`	numerical. Select out of: 1: layout.fruchterman.reingold(g) 2: layout.random(g) 3: layout.kamada.kawai(g) 4: layout.circle(g) 5: layout.reingold.tilford(g) 6: layout1 <- layout.sphere(g)

Examples

```
x<-list(
     c("Laura","Ingmar"),
     c("Peter","Renate","Ingmar","Andrea"),
     c("Nassim","Ingmar","Sergej"),
     c("Laura","Rike","Andra"),
     c("Marlene","Nassim","Christina","Sabine"),
     c("Bela","Ingmar","Mariola","Nassim"),
     c("Gloria","Kim","Olek","Bolek"))
igraph2(x,seed=2)
```

`image2video` *Convert an image sequence to a video or animated gif*

Description

requires ffmpeg for video conversion: https://www.ffmpeg.org/

Usage

```
image2video(
  folder = getwd(),
  i.format = "png",
  container = "mp4",
  fps = 8,
  last = 2,
  loop = 0
)
```

Arguments

`folder`	folder with image sequence
`i.format`	image format (png, jpeg, pdf only working for gif conversion)
`container`	movie container: mp4, mov, avi, gif
`fps`	frames per second used
`last`	length of last frame in seconds
`loop`	loop gif (0=True,1=False)
`cex`	text size

Examples

```
x<-sample(paste("group",LETTERS[1:5]),200,T)
ani.barplot(prop.table(table(x)),folder="ani.barplot",format="png")
image2video(folder="./ani.barplot",cont="mov",i.f="png")
```

`itemplot` *itemplot*

Description

Function to draw distribution of Likert scaled item with relative frequencies, mean, 1., 2. and 3. quartile.

Usage

```
itemplot(
  x,
  col = "lightblue",
  col2 = c("red", "blue"),
  labels = c(levels(x)[1], levels(x)[length(levels(x))]),
  main = "",
  dec = 1,
  legend = TRUE,
  decmean = 2,
  quant.type = 4,
  pt.cex = 1.2,
 values = round(c(mean(a, na.rm = T), median(a, na.rm = T), quantile(a, 0.25, type =
   quant.type, na.rm = T), quantile(a, 0.75, type = quant.type, na.rm = T), length(a)),
    2),
 legendtext = c(as.expression(bquote("Mean" == .(round(mean(as.numeric(x), na.rm = T),
   decmean)))), as.expression(bquote("Median" == .(median(as.numeric(x), na.rm = T)))),
    as.expression(bquote("1. Quartile" == .(quantile(as.numeric(x), 0.25, type =
    quant.type, na.rm = T)))), as.expression(bquote("3. Quartile" ==
    .(quantile(as.numeric(x), 0.75, type = quant.type, na.rm = T)))),
    as.expression(bquote("N "[included] == .(sum(table(x))))),
    as.expression(bquote("N "[missing] == .(sum(is.na(x))))),
  cex = 1
)
```

Arguments

`x`	a likert scaled factor variable
`col`	color of bars
`col2`	color for mean and median
`labels`	a vector of length 2 with low and high level labels
`main`	title
`dec`	decimal digits for percentage on bars
`legend`	Logical. If TRUE draws legend
`decmean`	decimal digits for mean diplayed in legend
`quant.type`	quantile type
`pt.cex`	median and mean point size
`values`	legend values
`legendtext`	legends text
`cex`	text size

Examples

```
x<-factor(sample(letters[1:6],100,T))
itemplot(x,main="itemplot()",labels=c("I fully\n disagree","I fully\n agree"),legend=F)
itemplot(x,main="itemplot()",labels=c("I fully\n disagree","I fully\n agree"))
```

`multi.line` *multi.line*

Description

Draw multiple regression or lowess lines

Usage

```
multi.line(
  x,
  y,
  group,
  main = "",
  axes = TRUE,
  xlab = "",
  ylab = "",
  xlim = range(x, na.rm = T),
  ylim = range(y, na.rm = T) + c(0, (max(y, na.rm = T) - min(y, na.rm = T))/10),
  points = TRUE,
  col = rainbow(length(levels(factor(group)))),
  pch = 1:length(levels(factor(group))),
  cex = 1,
  lines = TRUE,
  lty = 1:length(levels(factor(group))),
  lwd = 2,
  type = "line",
  locator = "top",
  horiz = TRUE,
  names = paste(levels(group), " ")
)
```

Arguments

`x`	independend metric variable
`y`	dependend metric variable
`group`	grouping variable
`main`	title
`axes`	draw axes
`xlab`	x axis label
`ylab`	y axis label
`xlim`	x axis limits
`ylim`	y axis limits
`points`	draw points
`col`	point color
`pch`	point character
`cex`	point size
`lines`	draw lines
`lty`	line type
`lwd`	line width
`type`	draw regression line/s or lowess line ("line", "lowess", "both")
`locator`	legend location
`horiz`	align horizontally
`names`	group names

Examples

```
data(ChickWeight)
attach(ChickWeight)
Diet<-paste("Diet",ChickWeight$Diet)
multi.line(x=jitter(Time,2),y=weight,group=Diet,points=T,main="multi.line()",xlab="time",ylab="weight")
multi.line(x=jitter(Time,2),y=weight,group=Diet,points=F,main="multi.line() without points with lowess line",xla
```

multireg — *Draws and estimates multiple regression model with two independent variables*

Description

Draws and estimates multiple regression model with two independent variables

Usage

```
multireg(
  y,
  x,
  z,
  interaction = TRUE,
  ylab = NULL,
  xlab = NULL,
  main = NULL,
  col = NULL,
  xlim = NULL,
  ylim = NULL,
  lwd = 2,
  pch = 16,
  legend = TRUE,
  locator = "top",
  lty = NULL,
  bty = "n",
  horiz = TRUE,
  title = NULL,
  shift = 0.15,
  quantiles_z = c(0.25, 0.5, 0.75),
  colquant = 2:(length(quantiles_z) + 1)
)
```

Arguments

`x` # a numeric or factor variable

`z` # a numeric or factor variable

`interaction=TRUE`
Logical. if TRUE calculates and draws model with interaction

`ylab=NULL` y axis label

`xlab=NULL` x axis label

`main=NULL` main title

`col=NULL` color

`xlim=NULL` limits of x axis

`ylim=NULL`	limits of x axis
`lwd=2`	regression line width
`pch=16`	point character
`legend=TRUE`	Logical, if TRUE draws legend
`locator="top"`	locatin of legend
`lty=NULL`	line type of legend elements
`bty="n"`	box type of legend
`horiz=TRUE`	adjust legend text horizontally
`title=NULL`	legend title
`shift=.15`	distance of groups in interactionplot
`quantiles_z=c(.25, .5, .75)`	vector of quantiles to split z
`colquant=2:(length(quantiles_z)+1)`	vector of colors for groups

Examples

```
# create variables
metricY<-rnorm(100)
metricX1<-rnorm(100)
metricX2<-rnorm(100)
binaryY<-rbinom(100,1,.5)
factorY<-factor(sample(paste("Category",1:3),100,T))
factorX1<-factor(sample(paste("Level X1",1:3),100,T))
factorX2<-factor(sample(paste("Level X2",1:3),100,T))
metricY<metricX1*metricX2+rnorm(100)
multireg(metricY,metricX1,metricX2)
binaryY<-ifelse(((metricX1*metricX2)+rnorm(100))>0,1,0)
# use multireg
multireg(metricY,metricX1,metricX2,legend=T,main="metric Y ~ metric X1 * metric X2")#
multireg(metricY,metricX1,factorX2,legend=T,title=NULL,main="metric Y ~ metric X1 * factor X2")#
multireg(metricY,factorX1,factorX2,legend=T,main="metric Y ~ factor X1 * factor X2")#
multireg(binaryY,metricX1,metricX2,legend=,main="binary Y ~ metric X1 * metric X2")#
multireg(binaryY,metricX1,factorX2,legend=F,title=NULL,main="binary Y ~ metric X1 * factor X2")#
multireg(binaryY,factorX1,factorX2,legend=T,main="binary Y ~ factor X1 * factor X2")#
multireg(factorY,metricX1,metricX2,legend=T,main="factor Y ~ metric X1 * metric X2")#
multireg(factorY,metricX1,factorX2,legend=T,main="factor Y ~ metric X1 * factor X2")#
multireg(factorY,factorX1,factorX2,legend=T,main="factor Y ~ factor X1 * factor X2")#
```

`ORCiD` *ORCIDs and the corresponding author names*

Description

A dataset containing close to 1,000,000 ORCIDs and corresponding names

Usage

```
ORCiD
```

Format

A data frame with close to 1,000,000 rows and 2 variables:

ID ORCID

Author Author name ...

Source

https://orcid.org/

`orcid2name` *orcid2name*

Description

Function to convert ORCiD to name with orcid.org api (needs a valid ORCID token for OAuth authentication)

Usage

```
orcid2name(x, useDB = TRUE, api = TRUE)
```

Arguments

`x`	vector that contains a valid ORCID-address (e.g.: "https://orcid.org/0000-0003-1159-3991") and/or names.
`useDB`	logical. IF TRUE a self curated list with ORCIDs from within the PMC database is beeing used before connecting to the API to gather still missing ORCIDs.
`api`	logical. IF TRUE the ORCID api is used to convert ORCiDs to names. Set to FALSE if no ORCiD authentification is possible.

Examples

```
orcid2name(c("https://orcid.org/0000-0003-1159-3991","Einstein, Albert"))
```

prettybarplot *prettybarplot*

Description

draws a barplot of a one or two dimensional frequency table with frequencies on top of bars and pretty background

Usage

```
prettybarplot(
  x,
  col = "standard",
  ylab = NULL,
  xlab = NULL,
  main = "",
  names = "",
  ylim = c(0, max(x) + max(x)/7),
  box = FALSE,
  space = 0.2,
  border = "black",
  text = TRUE,
  cex = 1,
  coltext = 1,
  pre = "n = ",
  post = "%",
  digit = 1,
  legend = TRUE,
  bg = TRUE,
  bg.increase = TRUE,
  bg.from = "grey",
  bg.to = "white",
  axes = TRUE,
  cex.axis = par("cex.axis"),
  cex.names = par("cex.axis"),
  las = 1
)
```

Arguments

`x`	a one or two dimensional frequency table
`col`	color of bars
`ylab`	y axis label
`xlab`	x axis label
`main`	title

`names`	name of bars/bar groups (in 2 dimensional table)
`ylim`	y-axis limits
`box`	draw box around plot
`space`	space between bars
`border`	border color of bars
`text`	draw text on bars
`cex`	text size
`coltext`	color of text on bars
`pre`	pre text for absolute frequencies
`post`	post text for relative frequencies
`digit`	digits to draw percentual frequencies with
`legend`	Logical. If TRUE draws legend
`bg`	draw background
`bg.increase`	top-down increase/decrease in background color
`bg.from`	background starting color
`bg.to`	background end color
`axes`	draw axes
`cex.axis`	size of bar labels
`cex.names`	size of axis labels
`las`	rotation of y limts (1: horizontal, 3: vertical)

Examples

```
x<-sample(paste("group",1:5),20000,T)
y<-sample(c("car","bike","bus"),20000,T)
# one dimensional barplot
prettybarplot(cumsum(table(x)))
prettybarplot(prop.table(table(x)))
# two dimensional barplot
prettybarplot(table(y,x))
prettybarplot(prop.table(table(y,x),m=2))
```

`profileline` *Profile line (of multiple groups) in multiple items*

Description

draws a profile line of means or medians in multiple items and groups

Usage

```
profileline(
  matrix,
  group = 1,
  type = "Median",
  groupnames = levels(factor(group)),
  main = "",
  col = 2:(length(unique(group)) + 1),
  pch = 1:length(levels(factor(group))),
  lty = 1:length(levels(factor(group))),
  lwd = 2,
  grid = TRUE,
  space_left = max(nchar(names(matrix)))/3 + 2.5,
  xlab = "",
  labels = "",
  extreme = FALSE,
  cex.axis.x = 1,
  cex.axis.y = 1,
  axis.top = TRUE,
  axis.bottom = TRUE,
  low_label = "",
  high_label = "",
  padj = 3,
  bottom.adj = 0,
  legend = TRUE,
  N_legend = TRUE,
  cex.legend = 1,
  title = TRUE
)
```

Arguments

`matrix` a data.frame or matrix containing the likert variables to draw

`group` grouping variable for profile lines (optional)

`type` type of line: one out of "Median", "Mean", "1. Quartile", "3. Quartile"

`groupnames` names of groups

`main` title

`col` color of group lines

`pch` point character

`lty` point character

`lwd` line width of profile line

`grid` draw grid

`space_left` space to y axis of first bar

`xlab` x axis label

`labels` labels displayed on x axes (must be of length

extreme	only draw poles of labels (senseful for likert items)
cex.axis.x	cex of x axis
cex.axis.y	cex of y axis
axis.top	draw top axis
axis.bottom	draw bottom axis
low_label	lowest level
high_label	highes level
padj	vertical adjust of low/high levels
bottom.adj	mar bottom adjust
legend	draw legend
N_legend	draw legend of N observations per group
cex.legend	character expansation of legend
title	legends title

Examples

```
matrix<-data.frame(matrix(sample(-3:3,1000,T),ncol=20))
names(matrix)<-paste("Item",1:20); g<-paste("Group",rep(1:3,length=200))
for(i in 1:dim(matrix)[2]){matrix[,i]<-factor(matrix[,i],-3:3,c("fully disagree","2","3","4","5","6","fully agre
profileline(matrix,g,main="",N_legend=T,pch="",type="Mean",ex=T,low="low",high="high",xlab="agreement")
```

stacked.area.chart *stacked area chart*

Description

stacked area chart for 2 dimensional frequency tables

Usage

```
stacked.area.chart(
  x,
  type = "area",
  col = NULL,
  xpos.text = NULL,
  reorder = FALSE,
  labels = TRUE,
  addFreq = TRUE,
  main = "",
  xlab = "",
  ylab = "",
  cex = 0.9,
  cex.axis = 1,
  las = 1,
  ylim = NULL
)
```

Arguments

`x`	2 dimensional frequency table
`type`	type of grapf ("area" or "line")
`col`	color of areas
`xpos.text`	x position of text
`xlab`	x axis label
`ylab`	y axis label
`cex`	character expansation of area labels
`cex.axis`	character expansation of axis labels
`las`	las for y axis labels
`ylim`	limits of x axis
`labels=TRUE`	Logicical. If TRUE labels of areas are drawn on the right side of the plot
`reorder=FALSE`	Logical. If TRUE reorders by frequency of y

Examples

```
x<-sample(2010:2020,100000,T,.2+(1:11)*.5)
y<-sample(paste("category",LETTERS[1:10]),100000,T)
par(mar=c(4,4,2,15))
stacked.area.chart(table(x,y),main="stacked.area.chart()",ylab="h(x)",xlab="year",reorder=T)
stacked.area.chart(prop.table(table(x,y),m=1),main="stacked.area.chart()",ylab="f(x)",xlab="year",reorder=T)
```

`worldmap2` *worldmap2*

Description

worldmap2

Usage

```
worldmap2(
  country,
  connect = TRUE,
  connections = NULL,
  reduce2country = NULL,
  region = NULL,
  main = "country and country connection frequency",
  ocean = "grey95",
  missingCol = "white",
  legend = TRUE,
  xpd = TRUE,
  mar = c(0, 0, 0, 0),
  mai = c(0, 0, ifelse(legend == TRUE, 0.1, 0), 0),
  palette = "Blues",
  legend1adjust = 0,
  legend2adjust = 0
)
```

Arguments

`country`	a list with vectors or a frequency table of country names
`connect`	logical. If TRUE draws country connections
`connections`	an optional vector with connections seperated with " - " (e.g.: "Germany - USA")
`reduce2country`	character. reduces connections to those of one specific country
`region`	one of: ('eurasia' 'africa' 'latin america' 'north america' 'uk' 'oceania' 'asia')
`main`	title
`ocean`	ocean color
`missingCol`	color of non present countries
`legend`	logical. If TRUE legends are drawn
`xpd`	logical. If TRUE ocean color is set to background color
`mar`	mar adjustment
`mai`	mai adjustment
`palette`	colour palette from brewer.pal(). One of 'Blues', 'BuGn', 'BuPu', 'GnBu', 'Greens', 'Greys', 'Oranges', 'OrRd', 'PuBu', 'PuBuGn', 'PuRd', 'Purples', 'RdPu', 'Reds', 'YlGn', 'YlGnBu', 'YlOrBr', 'YlOrRd', 'BrBG', 'PiYG', 'PRGn', 'PuOr', 'RdBu', 'RdGy', 'RdYlBu', 'RdYlGn' 'Spectral'
`legend1adjust`	adjust y position of upper legend and title
`legend2adjust`	adjust y position of lower legend

Examples

```
worldmap2(country=list(c("Germany","Austria","S. Korea"),
c("Germany","Domenican Rep.","S. Korea"),c("Germany","Austria","S. Korea"),
c("Germany","Spain"),c("Germany","Spain"),c("United States","Germany"),
c("Germany","Cuba","Bolivia","South Africa")),reduce2country="USA",
      #connections=list("NA"," - ",c("Germany - USA","Germany - Peru"),"Germany - USA","Germany - Austria","Germa
         legend1=-10,legend2=-15)
```

Index