Package ‘MatriKS’

March 9, 2023

Type Package

Title Generates Stimuli According to Rules

Version 0.1.0

Author Who wrote it

Maintainer The package maintainer `<yourself@somewhere.net>`

Description More about what it does (maybe more than one line)
Use four spaces when indenting paragraphs within the Description.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports DescTools

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

RoxygenNote 7.2.0

R topics documented:

axe *Define the cooordinates for an axe*

Description

Define the cooordinates for an axe

Usage

```
axe(size.x = 15, pos.x = 0, pos.y = 0, lty = 1, lwd = 3, shd = NA)
```

Arguments

`shd`

biscuit *Define the coordinates of a biscuit*

Description

Define the coordinates of a biscuit

Define the coordinates of a biscuit

Usage

```
biscuit(
  pos.x = 0,
  pos.y = 0,
  size.x = 10,
  size.y = 10,
  shd = "black",
  lty = 1,
  lwd = 3
)

biscuit(
  pos.x = 0,
  pos.y = 0,
  size.x = 10,
  size.y = 10,
  shd = "black",
  lty = 1,
  lwd = 3
)
```

Arguments

lty

lwd

Value

Based on the chosen shape, return an object with the information for plotting the desired design. If the name is precedeed by an s, the object is seen as a unique object, otherwise it is seen as a combination of multiple objects

Examples

```
One day
```

---

bow.tie *Define the coordinates of a vertical bow tie*

Description

Define the coordinates of a vertical bow tie

Define the cooordinates for a vertical bow tie

Usage

```
bow.tie(size.x = 10, pos.x = 0, shd = NA, lwd = 3, lty = 1)

bow.tie(size.x = 10, pos.x = 0, shd = NA, lwd = 3, lty = 1)
```

Arguments

lwd

lty

`bow.tie.inv` *Define the cooordinates for an horizontal bow tie*

Description

Define the cooordinates for an horizontal bow tie

Usage

```
bow.tie.inv(size.x = 10, pos.x = 0, shd = NA, lwd = 3, lty = 1)
```

Arguments

`lty`

`circle` *Define the coordinates of a circle*

Description

Define the coordinates of a circle

Usage

```
circle(
  size.x = 10,
  size.y = 10,
  pos.x = 0,
  pos.y = 0,
  shd = NA,
  rot = 0,
  vis = 1,
  lty = 1,
  lwd = 3
)
```

Arguments

`size.x`	Length of the x axis
`size.y`	Length of the y axis
`pos.x`	Position on the x axis
`pos.y`	Position on the y axis
`shd`	Color of the object. Deafsult is NA which results in a transparent object
`rot`	Rotation of the ellipse in which the figure is inscribed
`vis`	Integer, indicates whether the object should be visible (1) or not (0). Deafult is visible
`lty`	Border line. Default is 1 (solid), can be dotted (2) or dashed (3)
`lwd`	Width of the border line. Deafult is 3.

Value

Based on the chosen shape, return an object with the information for plotting the desired design

Examples

```
Ci sarà
```

`cof` *Concatenation of objects*

Description

Concatenation of objects

Usage

```
cof(..., name, single)
```

Arguments

`...`	Vector of objects to be concatened together.
`name`	Name of the newly created object (See Details).
`single`	Should the objects be collapsed into a single object? Deafult is FALSE

Value

An object of class cell

Examples

```
Poi li scrivo un attimo
```

`cross` *Define the coordinates of a cross*

Description

Define the coordinates of a cross

Usage

```
cross(
  size.x = sqrt(square()$size.x[[1]]^2/2),
  size.y = sqrt(square()$size.y[[1]]^2/2),
  lwd = 3,
  lty = 1
)
```

Arguments

`lty`

cross.dice *Define the coordinates of a cross dice with 4 dots*

Description

Define the coordinates of a cross dice with 4 dots

Usage

```
cross.dice(shd = "black", lwd = 3, lty = 1)
```

Arguments

lty

decof.cell *Split the elements of a cell*

Description

Split the elements of a cell

Usage

```
## S3 method for class 'cell'
decof(obj)
```

Arguments

The	object to be splitted

diagline *Define the coordinates of the main diagonal line*

Description

Define the coordinates of the main diagonal line

Usage

```
diagline(
  size.x = list(sqrt(square()$size.x[[1]]^2/2)),
  size.y = list(sqrt(square()$size.x[[1]]^2/2)),
  pos.x = 0,
  pos.y = 0,
  lty = 1,
  lwd = 3,
  rotation = pi - pi/4,
  vis = 1
)
```

Arguments

vis

diagline.inv *Define the coordinates of the secondary diagonal line*

Description

Define the coordinates of the secondary diagonal line

Usage

```
diagline.inv(
  size.x = sqrt(square()$size.x[[1]]^2/2),
  size.y = sqrt(square()$size.y[[1]]^2/2),
  pos.x = 0,
  pos.y = 0,
  lty = 1,
  lwd = 3,
  rotation = pi + pi/4,
  vis = 1
)
```

Arguments

vis

dice *Define the coordinates of a dice with 4 dots*

Description

Define the coordinates of a dice with 4 dots

Usage

```
dice(pos.x = 13, pos.y = 13, shd = "black", lwd = 3, lty = 1)
```

Arguments

lty

dot *Define the coordinates of a dot*

Description

Define the coordinates of a dot

Usage

```
dot(
  size.x = 2,
  size.y = 2,
  pos.x = 0,
  pos.y = 0,
  shd = "black",
  lty = lty,
  lwd = lwd,
  vis = 1
)
```

Arguments

`vis`

Examples

```
One day not today
```

down.petal *Define the coordinates of a the down petal*

Description

Define the coordinates of a the down petal

Usage

```
down.petal(lwd = 3, lty = 1)
```

Arguments

`lty`

draw *Draw objects*

Description

Draw objects

Usage

```
draw(
  obj,
  main = NULL,
  canvas = TRUE,
  hide = FALSE,
  n.cell = 9,
  bg = "white",
  mar = c(1, 1, 1, 1),
  xlim = 16
)
```

Arguments

obj	The object to be draw. Can be a single object, a matrix, or the responses
main	Print a title? Default is FALSE
canvas	Do you want to overimpose the objects? Default is FALSE
hide	Do you want to hide the cell of the correct response? Default is FALSE
n.cell	How main cell should the matrix have? Default is 9
bg	Choose the color of the background. Deafult is white
mar	Change margins
xlim	Change the length of the x axis

Value

A graphic

ellipse *Define the coordinates of an ellipse*

Description

Define the coordinates of an ellipse

Usage

```
ellipse(
  size.x = 10,
  size.y = 7,
  rot = 0,
  shd = NA,
  pos.x = 0,
  pos.y = 0,
  vis = 1,
  lty = 1,
  lwd = 3
)
```

Value

Return the default ellipse object

Examples

```
ellipse()
```

`flower` *Define the coordinates of a a flower*

Description

Define the coordinates of a a flower

Usage

```
flower(lwd = 3, lty = 1)
```

Arguments

`lty`

`h.arc.left.down` *Define the coordinates of the horizontal left down arch*

Description

Define the coordinates of the horizontal left down arch

Usage

```
h.arc.left.down(
  size.x = square()$size.x[[1]]/2,
  size.y = square()$size.y[[1]]/2,
  lty = 1,
  lwd = 3,
  vis = 1,
  pos.x = 0,
  pos.y = 0
)
```

Value

Return the horizontal arc left down object

Examples

```
h.arc.left.down()
```

h.arc.left.up *Define the coordinates of the horizontal left up arch*

Description

Define the coordinates of the horizontal left up arch

Usage

```
h.arc.left.up(
  size.x = square()$size.x[[1]]/2,
  size.y = square()$size.y[[1]]/2,
  lty = 1,
  lwd = 3,
  vis = 1,
  pos.x = 0,
  pos.y = 0
)
```

Value

Return the horizontal arc left up object

Examples

```
h.arc.left.up()
```

h.arc.right.down *Define the coordinates of the horizontal right down arch*

Description

Define the coordinates of the horizontal right down arch

Usage

```
h.arc.right.down(
  size.x = square()$size.x[[1]]/2,
  size.y = square()$size.y[[1]]/2,
  lty = 1,
  lwd = 3,
  vis = 1,
  pos.x = 0,
  pos.y = 0
)
```

Value

Return the horizontal arc right down object

Examples

```
h.arc.right.down()
```

h.arc.right.up *Define the coordinates of the horizontal right up arch*

Description

Define the coordinates of the horizontal right up arch

Usage

```
h.arc.right.up(
  size.x = square()$size.x[[1]]/2,
  size.y = square()$size.y[[1]]/2,
  lty = 1,
  lwd = 3,
  vis = 1,
  pos.x = 0,
  pos.y = 0
)
```

Value

Return the horizontal arc right up object

Examples

```
h.arc.right.up()
```

hello *Hello, World!*

Description

Prints ’Hello, world!’.

Usage

```
hello()
```

Examples

```
hello()
```

hexagon *Define the coordinates of an hexagon*

Description

Define the coordinates of an hexagon

Usage

```
hexagon(
  size.x = 15,
  size.y = 15,
  rot = 0,
  pos.x = 0,
  pos.y = 0,
  shd = NA,
  vis = 1,
  lty = 1,
  lwd = 3
)
```

Arguments

`lwd`

hide *Functions to modify the visible objects in a cell*

Description

Functions to modify the visible objects in a cell

Usage

```
hide(obj, index)
```

Arguments

`obj`	The cell of a matrix
`index`	The index of the element to hide/show/replace
`replacement`	The object with which an element should be replaced

Examples

```
Arrivano
```

hline *Define the coordinates of an horizontal line*

Description

Define the coordinates of an horizontal line

Usage

```
hline(
  size.x = sqrt(square()$size.x[[1]]^2/2),
  size.y = sqrt(square()$size.y[[1]]^2/2),
  pos.x = 0,
  pos.y = 0,
  lty = 1,
  lwd = 3,
  vis = 1
)
```

Arguments

`vis`

horizontal.eight *Define the coordinates of an horizontal eight*

Description

Define the coordinates of an horizontal eight

Usage

```
horizontal.eight(lwd = 3, lty = 1)
```

Arguments

`lty`

left.petal *Define the coordinates of a the left petal*

Description

Define the coordinates of a the left petal

Usage

```
left.petal(lwd = 3, lty = 1)
```

Arguments

`lty`

lily *Define the coordinates of a a lily*

Description

Define the coordinates of a a lily

Usage

```
lily(lwd = 3, lty = 1)
```

Arguments

`lty`

luck *Define the coordinates of a luck*

Description

Define the coordinates of a luck

Usage

```
luck(
  size.x = 10,
  size.y = 15,
  rot = pi/2,
  pos.x = 0,
  pos.y = 0,
  shd = NA,
  vis = 1,
  lty = 1,
  lwd = 3
)
```

Arguments

`lwd`

luck.4 *Define the coordinates of a luck composed of 4 lines*

Description

Define the coordinates of a luck composed of 4 lines

Usage

```
luck.4(size.x = 10, size.y = 7, lwd = 3, lty = 1)
```

Arguments

`lty`

malta *Define the cooordinates for a Malta cross*

Description

Define the cooordinates for a Malta cross

Usage

```
malta(size.x = 10, pos.x = 0, shd = NA, lwd = 3, lty = 1)
```

Arguments

`lty`

maxi *Define the cooordinates for a maxi*

Description

Define the cooordinates for a maxi

Usage

```
maxi(size.x = 8, size.y = 4, pos.x = 0, shd = NA, lty = 1, lwd = 3)
```

Arguments

`lwd`

ninja *Define the coordinates of a ninja star*

Description

Define the coordinates of a ninja star

Usage

```
ninja(size.x = 10, size.y = 15, shd = "black", lwd = 3, lty = 0)
```

Arguments

`lty`

pacman *Define the coordinates of a pacman*

Description

Define the coordinates of a pacman

Usage

```
pacman(
  size.x = sqrt(square()$size.x[[1]]^2/2),
  size.y = 0,
  pos.x = 0,
  pos.y = 0,
  theta1 = pi/4,
  theta2 = 7 * pi/4,
  lty = 1,
  lwd = 3,
  shd = NA,
  vis = 1
)
```

Arguments

`vis`

pentagon *Define the coordinates of a pentagon*

Description

Define the coordinates of a pentagon

Usage

```
pentagon(
  size.x = 15,
  size.y = 15,
  rot = pi/2,
  pos.x = 0,
  pos.y = 0,
  shd = NA,
  vis = 1,
  lty = 1,
  lwd = 3
)
```

Arguments

`lwd`

pizza.2 *Define the coordinates of a pizza with two slices*

Description

Define the coordinates of a pizza with two slices

Usage

```
pizza.2(
  size.x = 15,
  size.y = 0,
  pos.x = 0,
  pos.y = 0,
  shd = NA,
  lty = 1,
  lwd = 3
)
```

Arguments

`lwd`

pizza.2.rev *Define the coordinates of an inverse pizza with two slices*

Description

Define the coordinates of an inverse pizza with two slices

Usage

```
pizza.2.rev(
  size.x = 15,
  size.y = 0,
  pos.x = 0,
  pos.y = 0,
  shd = NA,
  lty = 1,
  lwd = 3
)
```

Arguments

`lwd`

`pizza.4` *Define the coordinates of a pizza with four slices*

Description

Define the coordinates of a pizza with four slices

Usage

```
pizza.4(size.x = 15, shd = NA, lwd = 3, lty = 1)
```

Arguments

`lty`

`rectangle` *Define the coordinates of a rectangle*

Description

Define the coordinates of a rectangle

Usage

```
rectangle(
  size.x = 15,
  size.y = 20,
  rot = pi/4,
  pos.x = 0,
  pos.y = 0,
  shd = NA,
  lwd = 3,
  lty = 1,
  vis = 1
)
```

Arguments

`vis`

right.petal *Define the coordinates of a the right petal*

Description

Define the coordinates of a the right petal

Usage

```
right.petal(lwd = 3, lty = 1)
```

Arguments

`lty`

rotation *Title*

Description

Title

Usage

```
rotation(obj, n, rule, ...)
```

Arguments

`...`

Examples

```
onje day not todaty
```

s.axe *Define the cooordinates for a single axe (to be used in diff_shapes)*

Description

Define the cooordinates for a single axe (to be used in diff_shapes)

Usage

```
s.axe(size.x = 15, pos.x = 0, pos.y = 0, lty = 1, lwd = 3, shd = NA)
```

Arguments

`shd`

`s.biscuit` *Define the coordinates of a biscuit (to be used in diff_shapes)*

Description

Define the coordinates of a biscuit (to be used in diff_shapes)

Usage

```
s.biscuit(
  pos.x = 0,
  pos.y = 0,
  size.x = 10,
  size.y = 10,
  shd = "black",
  lty = 1,
  lwd = 3
)
```

Arguments

`lwd`

`s.bow.tie` *Define the coordinates of a single vertical bow tie (to be used in diff_shapes)*

Description

Define the coordinates of a single vertical bow tie (to be used in diff_shapes)

Define the cooordinates for a single bow tie (to be used in diff_shapes)

Usage

```
s.bow.tie(size.x = 10, pos.x = 0, shd = NA, lwd = 3, lty = 1)
```

```
s.bow.tie(size.x = 10, pos.x = 0, shd = NA, lwd = 3, lty = 1)
```

Arguments

`lwd`

`lty`

s.bow.tie.inv *Define the cooordinates for a single bow tie inverse (to be used in diff_shapes)*

Description

Define the cooordinates for a single bow tie inverse (to be used in diff_shapes)

Usage

```
s.bow.tie.inv(size.x = 10, pos.x = 0, shd = NA, lwd = 3, lty = 1)
```

Arguments

`lty`

s.flower *Define the coordinates of a single flower (to be used in diff shapes)*

Description

Define the coordinates of a single flower (to be used in diff shapes)

Usage

```
s.flower(lwd = 3, lty = 1)
```

Arguments

`lty`

s.horizontal *Define the cooordinates for an horizontal s*

Description

Define the cooordinates for an horizontal s

Usage

```
s.horizontal(lty = 1, lwd = 3)
```

Arguments

`lwd`

`s.horizontal.eight` *Define the coordinates of a single horizontal eight (to be used in diff_shapes)*

Description

Define the coordinates of a single horizontal eight (to be used in diff_shapes)

Usage

```
s.horizontal.eight(lwd = 3, lty = 1)
```

Arguments

`lty`

`s.horizontal.inv` *Define the cooordinates for an horizontal invertred s*

Description

Define the cooordinates for an horizontal invertred s

Usage

```
s.horizontal.inv(lty = 1, lwd = 3)
```

Arguments

`lwd`

`s.lily` *Define the coordinates of a a single lily (to be used in diff_shapes)*

Description

Define the coordinates of a a single lily (to be used in diff_shapes)

Usage

```
s.lily(lwd = 3, lty = 1)
```

Arguments

`lty`

s.malta *Define the cooordinates for a single Malta cross (to be used in diff_shapes)*

Description

Define the cooordinates for a single Malta cross (to be used in diff_shapes)

Usage

```
s.malta(size.x = 10, pos.x = 0, shd = NA, lwd = 3, lty = 1)
```

Arguments

`lty`

s.maxi *Define the cooordinates for a single maxi (to be used in diff_shapes)*

Description

Define the cooordinates for a single maxi (to be used in diff_shapes)

Usage

```
s.maxi(size.x = 8, size.y = 4, pos.x = 0, shd = NA, lty = 1, lwd = 3)
```

Arguments

`lwd`

s.pizza.2 *Define the coordinates of a single pizza with two slices (to be used in diff_shapes)*

Description

Define the coordinates of a single pizza with two slices (to be used in diff_shapes)

Usage

```
s.pizza.2(
  size.x = 15,
  size.y = 0,
  pos.x = 0,
  pos.y = 0,
  shd = NA,
  lty = 1,
  lwd = 3
)
```

Arguments

`lwd`

`s.pizza.2.rev` *Define the coordinates of a single inverse pizza with two slices (to be used in diff_shapes)*

Description

Define the coordinates of a single inverse pizza with two slices (to be used in diff_shapes)

Usage

```
s.pizza.2.rev(
  size.x = 15,
  size.y = 0,
  pos.x = 0,
  pos.y = 0,
  shd = NA,
  lty = 1,
  lwd = 3
)
```

Arguments

`lwd`

`s.pizza.4` *Define the coordinates of a single pizza with four slices*

Description

Define the coordinates of a single pizza with four slices

Usage

```
s.pizza.4(size.x = 15, shd = NA, lwd = 3, lty = 1)
```

Arguments

`lty`

s.s.horizontal *Define the cooordinates for a single horizontal s (to be used in diff shapes)*

Description

Define the cooordinates for a single horizontal s (to be used in diff shapes)

Usage

```
s.s.horizontal(lty = 1, lwd = 3)
```

Arguments

`lwd`

s.s.horizontal.inv *Define the cooordinates for a single inverted horizontal s (to be used in diff shapes)*

Description

Define the cooordinates for a single inverted horizontal s (to be used in diff shapes)

Usage

```
s.s.horizontal.inv(lty = 1, lwd = 3)
```

Arguments

`lwd`

s.s.vertical *Define the cooordinates for a single vertical s (to be used in diff_shapes)*

Description

Define the cooordinates for a single vertical s (to be used in diff_shapes)

Usage

```
s.s.vertical(lty = 1, lwd = 3)
```

Arguments

`lwd`

s.s.vertical.inv *Define the cooordinates for a single inverted vertical s (to be used in diff_shapes)*

Description

Define the cooordinates for a single inverted vertical s (to be used in diff_shapes)

Usage

```
s.s.vertical.inv(lty = 1, lwd = 3)
```

Arguments

`lwd`

s.vertical *Define the cooordinates for a vertical s*

Description

Define the cooordinates for a vertical s

Usage

```
s.vertical(lty = 1, lwd = 3)
```

Arguments

`lty`

Examples

```
Arrivano
```

s.vertical.eight *Define the coordinates of a single vertical eight (to be used in diff_shapes)*

Description

Define the coordinates of a single vertical eight (to be used in diff_shapes)

Usage

```
s.vertical.eight(lwd = 3, lty = 1)
```

Arguments

`lty`

s.vertical.inv *Define the cooordinates for a vertical inverted s*

Description

Define the cooordinates for a vertical inverted s

Usage

```
s.vertical.inv(lty = 1, lwd = 3)
```

Arguments

lwd

semi.circle *Define the coordinates of a downward-facing left semi-circle*

Description

Define the coordinates of a downward-facing left semi-circle

Usage

```
semi.circle(
  size.x = sqrt(square()$size.x[[1]]^2/2),
  size.y = 0,
  pos.x = 0,
  pos.y = 0,
  theta1 = pi/4,
  theta2 = 5 * pi/4,
  shd = NA,
  rotation = pi - pi/4,
  lty = 1,
  lwd = 3,
  vis = 1
)
```

Arguments

vis

`semi.circle.inv` *Define the coordinates of an upward-facing right semi-circle*

Description

Define the coordinates of an upward-facing right semi-circle

Usage

```
semi.circle.inv(
  size.x = sqrt(square()$size.x[[1]]^2/2),
  size.y = 0,
  pos.x = 0,
  pos.y = 0,
  theta1 = 5 * pi/4,
  theta2 = pi/4,
  shd = NA,
  rotation = pi - pi/4,
  lty = 1,
  lwd = 3,
  vis = 1
)
```

Arguments

`vis`

`semi.circle.rev` *Define the coordinates of an downward facing right semi-circle*

Description

Define the coordinates of an downward facing right semi-circle

Usage

```
semi.circle.rev(
  size.x = sqrt(square()$size.x[[1]]^2/2),
  size.y = 0,
  pos.x = 0,
  pos.y = 0,
  theta1 = 7 * pi/4,
  theta2 = 3 * pi/4,
  shd = NA,
  rotation = pi - pi/4,
  lty = 1,
  lwd = 3,
  vis = 1
)
```

Arguments

vis

---

semi.circle.rev.inv *Define the coordinates of an upward facing left semi-circle*

Description

Define the coordinates of an upward facing left semi-circle

Usage

```
semi.circle.rev.inv(
  size.x = sqrt(square()$size.x[[1]]^2/2),
  size.y = 0,
  pos.x = 0,
  pos.y = 0,
  theta1 = 3 * pi/4,
  theta2 = 7 * pi/4,
  shd = NA,
  rotation = pi - pi/4,
  lty = 1,
  lwd = 3,
  vis = 1
)
```

Arguments

vis

---

slice *Define the coordinates of a pizza slice*

Description

Define the coordinates of a pizza slice

Usage

```
slice(
  size.x = 15,
  size.y = 0,
  pos.x = 0,
  pos.y = 0,
  theta1 = pi/4,
  theta2 = 3 * pi/4,
  lty = 1,
  lwd = 3,
  vis = 1,
  shd = NA
)
```

Arguments

shd

split.mat *Isolate ONLY the visible objects in a cell*

Description

Isolate ONLY the visible objects in a cell

Usage

```
## S3 method for class 'mat'
split(m, cell = NULL, vis = NULL, mat.type = 9)
```

Arguments

m	The matrix
cell	the index of the cell to be splitted
vis	boh
mat.type	Does the matrix have 4 or 9 cells? Default is 9

square *Define the coordinates of a square*

Description

Define the coordinates of a square

Usage

```
square(
  size.x = 15,
  size.y = size.x,
  rot = pi/4,
  pos.x = 0,
  pos.y = 0,
  shd = NA,
  vis = 1,
  lty = 1,
  lwd = 3
)
```

Value

Return the default square object

Examples

```
square()
```

square4 *Define the coordinates of a square composed of 4 lines*

Description

Define the coordinates of a square composed of 4 lines

Usage

```
square4(
  size.x = sqrt(square()$size.x[[1]]^2/2),
  size.y = sqrt(square()$size.y[[1]]^2/2),
  pos.x = sqrt(square()$size.x[[1]]^2/2),
  pos.y = sqrt(square()$size.y[[1]]^2/2),
  lwd = 3,
  lty = 1
)
```

Arguments

`lty`

star *Define the coordinates of a star*

Description

Define the coordinates of a star

Usage

```
star(size.x = 10, size.y = 15, shd = "black", lwd = 3, lty = 0)
```

Arguments

`lty`

`triangle` *Define the coordinates of a triangle*

Description

Define the coordinates of a triangle

Usage

```
triangle(
  size.x = 15,
  size.y = 15,
  pos.x = 0,
  pos.y = 0,
  rot = pi/2,
  shd = NA,
  vis = 1,
  lty = 1,
  lwd = 3
)
```

Value

Return the default triangle object

Examples

```
triangle()
```

`up.petal` *Define the coordinates of a the up petal*

Description

Define the coordinates of a the up petal

Usage

```
up.petal(lwd = 3, lty = 1)
```

Arguments

`lty`

v.arc.left.down *Define the coordinates of the vertical left down arch*

Description

Define the coordinates of the vertical left down arch

Usage

```
v.arc.left.down(
  size.x = square()$size.x[[1]]/2,
  size.y = square()$size.y[[1]]/2,
  lty = 1,
  lwd = 3,
  vis = 1,
  pos.x = 0,
  pos.y = 0
)
```

Value

Return the vertical arc left down object

Examples

```
v.arc.left.down()
```

v.arc.left.up *Define the coordinates of the vertical left up arch*

Description

Define the coordinates of the vertical left up arch

Usage

```
v.arc.left.up(
  size.x = square()$size.x[[1]]/2,
  size.y = square()$size.y[[1]]/2,
  pos.x = 0,
  pos.y = 0,
  vis = 1,
  lty = 1,
  lwd = 3
)
```

Arguments

`pos.y`

v.arc.right.down *Define the coordinates of the vertical right down arch*

Description

Define the coordinates of the vertical right down arch

Usage

```
v.arc.right.down(
  size.x = square()$size.x[[1]]/2,
  size.y = square()$size.y[[1]]/2,
  lty = 1,
  lwd = 3,
  vis = 1,
  pos.x = 0,
  pos.y = 0
)
```

Value

Return the vertical arc right down object

Examples

```
v.arc.right.down()
```

v.arc.right.up *Define the coordinates of the vertical right up arch*

Description

Define the coordinates of the vertical right up arch

Usage

```
v.arc.right.up(
  size.x = square()$size.x[[1]]/2,
  size.y = square()$size.y[[1]]/2,
  pos.x = 0,
  pos.y = 0,
  vis = 1,
  lty = 1,
  lwd = 3
)
```

Value

Return the vertical arc right up object

Examples

```
v.arc.right.up()
```

`vertical.eight` *Define the coordinates of a vertical eight*

Description

Define the coordinates of a vertical eight

Usage

```
vertical.eight(lwd = 3, lty = 1)
```

Arguments

`lty`

`vline` *Define the coordinates of a vertical line*

Description

Define the coordinates of a vertical line

Usage

```
vline(
  size.x = sqrt(square()$size.x[[1]]^2/2),
  size.y = sqrt(square()$size.y[[1]]^2/2),
  pos.x = 0,
  pos.y = 0,
  lty = 1,
  lwd = 3,
  vis = 1
)
```

Arguments

`vis`

`X` *Define the coordinates of an X*

Description

Define the coordinates of an X

Usage

```
X(
  size.x = sqrt(square()$size.x[[1]]^2/2),
  size.y = sqrt(square()$size.y[[1]]^2/2),
  lwd = 3,
  lty = 1
)
```

Arguments

`lty`

Index