Package: Vennerable (via r-universe)

May 21, 2026

Title Venn and Euler area-proportional diagrams

Version 3.1.0.9000

Author Jonathan Swinton

Description A package for drawing various species of Venn diagrams.

Maintainer Jonathan Swinton <jonathan@swintons.net>

LazyLoad yes

Depends methods, R (>= 2.10)

Imports graph, RBGL, grid, lattice, RColorBrewer, reshape2, gtools

URL https://github.com/js229/Vennerable

Suggests xtable

License GPL

Config/pak/sysreqs libicu-dev

Repository https://bedapub.r-universe.dev

Date/Publication 2016-03-14 16:32:19 UTC

RemoteUrl https://github.com/bedapub/Vennerable

RemoteRef HEAD

RemoteSha 46057c99ede68bfa8b937a6d0a990291e729e1c8

Contents

Vennerable-package 2
compute.S4 3
compute.Venn 4
PlotDarkMatter 5
PlotVennGeometry 5
StemCell 6
TissueDrawing-class 7
Venn 9
Venn-class 10
VennDiagrams 10

VennDrawing-class 11
Vennerable-internal 12
VennSetSetLabels 13
VennThemes 14

Index **16**

`Vennerable-package` *Venn and Euler area-proportional diagrams*

Description

A package for drawing various species of Venn diagrams.

Details

Package:	Vennerable
Version:	1.0
LazyLoad:	yes
Depends:	graph, RBGL, grid, lattice, RColorBrewer, reshape
License:	GPL
Built:	R 2.9.0; ; 2009-07-09 13:52:22 UTC; windows

Index:

```
Venn                    Construct intersections of sets
Venn-class              Class "Venn" ~~~
```

Further information is available in the following vignettes:

`Venn` (source, pdf)

Author(s)

Jonathan Swinton

Maintainer: Jonathan Swinton <jonathan@swintons.net>

compute.S4 *Venn diagrams on 4 sets*

Description

Given a Venn object on 4 sets, returns an VennDrawing object rendering it in the Lewis Carroll style.

Usage

```
compute.S4(V, doWeights = FALSE, s = 0.25, likeSquares = TRUE)
```

Arguments

V	A Venn object on 4 sets.
doWeights	Must be FALSE. For weighted diagrams, use compute.CR.
s	If $s = 0$ then set boundaries may be superimposed; for $s > 0$ the larger two rectangles are offset by s.
likeSquares	If TRUE, make the individual faces as square as possible; if FALSE remove additional corners and use triangular edges.

Details

The placement of the face names is not ideal for some combinations with $s > 0$.

Value

An object of class VennDrawing

Author(s)

Jonathan Swinton (jonathan@swintons.net)

Examples

```
compute.S4(Venn(n=4))
```

compute.Venn *Calculate the geometry of a Venn or Euler diagram.*

Description

Given an object of class Venn, compute an appropriate diagram.

Usage

```
compute.Venn(V, doWeights = TRUE, doEuler = FALSE, type)
```

Arguments

V	An object of class Venn
doWeights	Logical
doEuler	Logical
type	One of "circles","squares","triangles","ellipses","AWFE","ChowRuskey","AWFE".

Details

For more details see the package vignette with vignette("Venn")

Value

An object of class VennDrawing

Author(s)

Jonathan Swinton (jonathan@swintons.net)

Examples

```
setList <- strsplit(month.name,split="")
names(setList) <- month.name
compute.Venn(Venn(setList[1:3]))
```

PlotDarkMatter *Plots dark matter*

Description

Fills the area of a VennDrawing universe that is not occupied by any set.

Usage

```
PlotDarkMatter(VD)
```

Arguments

VD	An object of class VennDrawing

Details

This works by filling the entire universe with grey (not currently changeable) and then plotting then filling the inside of the dark matter boundary with white.

Value

Executed for its side effecrs

Author(s)

Jonathan Swinton (jonathan@swintons.net)

Examples

```
VD <- compute.Venn(Venn(n=2))
PlotDarkMatter(VD)
```

PlotVennGeometry *Display computed Venn diagrams*

Description

This is the function called by the plot generic for objects of class VennDrawing.

Usage

```
PlotVennGeometry(C3, gpList, show = list(FaceText = "weight"))
```

Arguments

C3	An object of class VennDrawing, as produced by compute.Venn
gpList	A list of lists of graphical parameters. The VennThemes function produces a suitable list; see its man page for more details.
show	A list of options controlling what is shown. Defaults are universe = TRUE, Sets = TRUE, SetLabels = TRUE, DarkMatter = FALSE, Faces = TRUE, FaceText = "weight". If universe = TRUE the bounding region, currently always a rectangle, of the diagram is shown. If Sets = TRUE, the boundaries of the Sets are shown. If SetLabels = TRUE the names of the Sets are plotted according to the instructions in VennGetSetLabels(C3). If Faces = TRUE individual faces are filled. FaceText is a character vector which may contain any of c("weight","signature","sets","elements"). Currently DarkMatter must be FALSE.

Details

The algorithm for displaying face annotation is pretty crude. The annotation corresponding to each member of the FaceText vector is displayed on a single line. weight and signature display the associated weight and signature for the face. sets produces a concatenated version of the names of the sets in the face, while elements does the same for the elements; both of these only really work for single character set names or elements.

Author(s)

Jonathan Swinton (jonathan@swintons.net)

Examples

```
library(Vennerable)
setList <- strsplit(month.name,split="")
names(setList) <- month.name
C3 <- compute.Venn(Venn(setList[1:3]))
PlotVennGeometry(C3,show=list(FaceText="signature"),gp=VennThemes(C3,colourAlgorithm="signature"))
```

StemCell *Stem cell data*

Description

Data on the genes which bind to each of three transcription factors (and a fourth which may also be a transcription factor)

Usage

```
data(StemCell)
```

Format

A named list with four elements named OCT4, SOX2, NANOG and E2F4. Each element is a character vector containing the names of 600-1700 associated genes.

Source

Taken from supplementary information in Boyer et al, "Core transcriptional regulatory circuitry in human embryonic stem cells", *Cell*, 2005, 122:947–956, 10.1016/j.cell.2005.08.020,http://www.ncbi.nlm.nih.gov/pubmed/16153702.

Examples

```
library(Vennerable)
data(StemCell)
plot(Venn(Sets=StemCell),type="squares",doWeights=FALSE)
```

TissueDrawing-class *Class "TissueDrawing"*

Description

A class used internally to implement polygon-polgyon clipping

Objects from the Class

The class describes diagrams which are constructed by superposing simple polygons (ie those with no holes or self-intersections), called Sets. If the Sets are filled with semi-transparent colours, the resulting diagram will have different regions or Faces depending on the intersections between the sets. The boundaries of these regions will not necessarily be simple. (For example one circle may have two other circles nested inside it). A simple model for this process is superpsosing Sets made out of tissue paper, hence the name of the class, which was nearly called a MAGDrawing after the perspex and lightbox exhibit in the children's gallery of the Manchester Art Gallery. which also acted as a prototype.

Points where the Sets intersect are always identified and named within the object. Sets which partially share a common boundary are defined not to intersect on that common boundary, so there are only a finite number of intersection points. Each Face contains at least one named point so additional named points will be introduced for each Face that has no other intersections. In addition, named points may be introduced to make some of the hidden-line alogorithms easier. These additional named points can be removed (sometimes) with the function remove.nonintersectionpoints which leaves the remaining points as the nodes of the graph defined by the Set intersections.

Set and Face boundaries are made up of named edges, each of which starts and finishes at the named points. Each edge can be either a polygonal segment or a sector of a circle. All of the Faces in a TissueDrawing are connected to each other (ie the graph is connected), if necessary by adding invisible edges which are not part of the boundary of any set.

Objects can be created by calls to the functions newTissueFromPolygon() or newTissueFromCircle, and combined with the function addSetToDrawing).

Slots

`setList`: Named `"list"`, with names given by the name of each Set. Each entry is a character vector of edge names

`nodeList`: Named `"list"`, with a 1x2 numeric matrix of (x,y) coordinates for each named point.

`recentChanges`: Nasty bodge slot required by internal algorithms

`edgeList`: A `TDEdgeList`, a named list of `TDedgeDrawn` objects defining the geometry of each edge

`faceList`: A `TDFaceList`, a named `"list"`, with one element for each correspondingly named face, listing the edges of that face in clockwise order

`faceSignature`: Named `"list"`, with names the same as `faceList`, giving the signature for the face.

Extends

Class `"TDEdgeList"`, directly. Class `"TDFaceList"`, directly.

Methods

PlotFaces `signature(drawing = "TissueDrawing")`: Displays the faces of the drawing on the current graphics device

PlotNodes `signature(drawing = "TissueDrawing")`: Displays the named points of the drawing

PlotSetBoundaries `signature(drawing = "TissueDrawing")`: Displays the Set boundaries of the drawing

show `signature(object = "TissueDrawing")`: Prints a summary of the drawing

Note

Adding even two overlapping simple polygons together can result in multiple intersections regions. Finding the boundaries of these intersections, keeping track of which have come from which combination of sets, and, in particular, coping with faces embedded inside outher faces is the problem of polygon-polygon clipping. This class is designed to solve that problem for the combinations of Sets used in the `Vennerable` package. It is not designed as a general clipping tool, and can be very slow, particular for clipping of general polygons. For example, the four ellipse plot is currently implemented as a combination of many short-segment polygons and several minutes to run.

Care is taken to keep track of the outside boundary of all of the sets, or equivalently the inside boundary of the 'dark matter', the Face defined as the region where none of the Sets intersect. Nevertheless this portion of the code is currently quite buggy and this can lead to errors in some computed diagrams.

Author(s)

Jonathan Swinton, jonathan@swintons.net

Examples

```
showClass("TissueDrawing")
```

Venn *Construct intersections of sets*

Description

Given a collection of Sets, Venn will compute all possible combinations of intersections and return an object of class Venn, storing the combinations as well as the number of elements in each intersection (the 'weights'). It will also accept a specification of the weights directly, or simply of the number of sets desired.

Usage

```
Venn(Sets, Weight, SetNames, numberOfSets)
Weights(object)
Weights(object) <- value
```

Arguments

Sets	A list, each element of which can be a list defining the members of a Set
Weight	A named vector of weights to associate with each possible combination of Set intersections. Ignored if Sets is present.
SetNames	A character vector of names for each Set. Ignored if Sets present and has names
numberOfSets	An integer for the number of Sets. Ignored if Sets or SetNames present.
object	An object of class Venn
value	An object like Weight

Value

An object of class Venn

Author(s)

Jonathan Swinton (jonathan@swintons.net)

Examples

```
Venn(n=3)
data(StemCell)
w <- Venn(Sets=StemCell[1:2])
Weights(w)
# nb order of intersection subsets not guaranteed
Weights(w) <- 1:4
Venn(SetNames=letters[1:5])
```

Venn-class *Class "Venn"*

Description

A class for intersecting multiple sets together

Objects from the Class

Objects should be created by calls to the function Venn

Slots

IndicatorWeight: Object of class "matrix"

IntersectionSets: Object of class "list". This may be empty.

Methods

[signature(x = "Venn"): ...

NumberOfSets signature(object = "Venn"): ...

plot signature(x = "Venn", y = "missing"): ...

SetNames signature(object = "Venn"): ...

show signature(object = "Venn"): ...

Author(s)

Jonathan Swinton (jonathan@swintons.net)

Examples

```
showClass("Venn")
```

VennDiagrams *Predrawn Venn diagrams.*

Description

Precomputed data on the geometrical structure of various types of Venn diagrams.

Usage

```
data(VennDiagrams)
```

Format

A list with elements named battle,AWFEscale,AWFE, and ellipses. Each element is itself a list in which the nth element, if not NULL, is an object of class TissueDrawing representing an example of an unweighted Venn diagram on n sets of the given type.

Details

This is really a device to cache the diagrams and is not for direct use. Use eg compute.Venn(Venn(n=...),type=... which returns a plottable object of class VennDrawing instead.

Source

Created by the buildVennDiagrams() function within Vennerable

Examples

```
data(VennDiagrams)
TD <- VennDiagrams[["ellipses"]][[4]]
class(TD)
```

VennDrawing-class *Class "VennDrawing"*

Description

A geometrical representation of a Venn diagram suitable for display

Objects from the Class

Objects can be created by calls of the form new("VennDrawing",V,TD) where V is an object of class Venn and TD is an object of class TissueDrawing which has representations for all the Sets and Faces corresponding to the sets and intersections of the V object.

As well as bringing together the geometric and set-theoretic properties of the Venn diagram, this class stores information about the universe in which to display the diagram and provides a number of methods for plotting the final figure.

Slots

universe: A matrix with the x and y coordinates of the bounding universe. Currently assumed to have 2 rows by much of the code, corresponding to the lower left and upper right corners of a bounding rectangle

SetLabels: A data.frame encoding the positions of labels for the Sets

FaceLabels: A data.frame encoding the positions of labels for the Faces

setList: See TissueDrawing-class

nodeList: See TissueDrawing-class

recentChanges: See TissueDrawing-class

edgeList: See TissueDrawing-class

faceList: See TissueDrawing-class

faceSignature: See TissueDrawing-class

IndicatorWeight: See Venn-class

IntersectionSets: See Venn-class

Extends

Class "TissueDrawing", directly. Class "Venn", directly. Class "TDEdgeList", by class "TissueDrawing", distance 2. Class "TDFaceList", by class "TissueDrawing", distance 2.

Methods

Areas signature(object = "VennDrawing"): Returns the area of each named face

IntersectionMidpoints signature(object = "VennDrawing"): ...

plot signature(x = "VennDrawing", y = "missing"): ...

PlotUniverse signature(object = "VennDrawing"): ...

show signature(object = "VennDrawing"): ...

UniverseRange signature(object = "VennDrawing"): ...

VisibleRange signature(object = "VennDrawing"): ...

Author(s)

Jonathan Swinton (jonathan@swintons.net)

Examples

```
showClass("VennDrawing")
```

Vennerable-internal *Internal functions for Vennerable, plus some user-visible ones I haven't documented yet*

Description

As well as a number of internal helper functions, Vennerable includes code for doing polygon-polygon clipping in some (buggy) generality which may be documented if there is demand.

Author(s)

Jonathan Swinton (jonathan@swintons.net)

VennSetSetLabels *Set and get annotation labels and graphical styles*

Description

Embedded within a VennDiagram object are instructions for displaying annotation for the sets and their faces. Retrieve them so they can be edited and re-embedded for subsequent display.

Usage

```
VennGetSetLabels(object)
VennSetSetLabels(object, SetLabels)
VennGetFaceLabels(object)
VennSetFaceLabels(object, FaceLabels)
VennGetUniverseRange(object)
VennSetUniverseRange(object, universe)
```

Arguments

object	An object of class VennDrawing
SetLabels	A data.frame. The format of this data.frame may change.
FaceLabels	A data.frame. The format of this data.frame may change.
universe	An $nx2$ matrix defining x and y coordinates of the bounding universe. Currently assumed to have 2 rows by much of the code, corresponding to the lower left and upper right corners of a bounding rectangle

Author(s)

Jonathan Swinton (jonathan@swintons.net)

Examples

```
data(StemCell)
Vstem <- Venn(StemCell)
Vstem3 <- Vstem[,c("OCT4","SOX2","NANOG")]
Cstem3 <- compute.Venn(Vstem3,doWeights=TRUE)
plot(Cstem3)
# don't like the default position of 'SOX2'?
SetLabels <- VennGetSetLabels(Cstem3)
SetLabels[SetLabels$Label=="SOX2","x"] <- 12
Cstem3 <- VennSetSetLabels(Cstem3,SetLabels)
grid::grid.newpage()
plot(Cstem3)
```

VennThemes *Create lists of graphical parameters for Venn diagrams*

Description

Given a VennDrawing object, which it consults to find the names of each of the sets and faces in the drawing, returns a list suitable as the gp argument in a subsequent call to the VennDrawing method for plot.

Usage

```
VennThemes(drawing, colourAlgorithm, increasingLineWidth)
```

Arguments

drawing	An object of class VennDrawing
colourAlgorithm	Missing or one of signature,binary,sequential.
increasingLineWidth	Logical, defaul FALSE

Details

Set boundary colours are taken from the Set1 palette provided by the RColorBrewer package. If colourAlgorithm="signature", face fill colours are taken frome the RColorBrewer YlOrRed palette based on the number of sets represented in the face, so eg all the faces corresponding to membership of a single set are pale yellow while the face corresponding to all the intersections is dark red. If colourAlgorithm="binary", faces are blue if they correspond to an odd number of intersections and white otherwise. If colourAlorithm="sequential", each face is given a different colour from the RColorBrewer Set3 palette, although this is repeated if necessary if there are more faces than the length of this palette (9). Different faces with the same signature will be given the same colour. If not specified, sequential is used if there are less than 9 faces, otherwise signature.

If increasingLineWidth=TRUE, each Set is given a different linewidth, with the last to be plotted given the thinnest width, to help in visualising nonsimple Venn diagrams.

Value

A list with four elements

Face	Named list of graphical parameters which will be applied to faces with corresponding names
FaceText	Named list of graphical parameters which will be applied to annotation in faces with corresponding names
Set	Named list of graphical parameters which will be applied to sets with corresponding names

FaceText — Named list of graphical parameters which will be applied to annotation in sets with corresponding names

These are graphical parameters in the sense of the grid package.

Author(s)

Jonathan Swinton (jonathan@swintons.net)

See Also

See also RColorBrewer

Examples

```
# change the name of one set  to red text and enlarge the other
C2 <- compute.Venn(Venn(n=2))
gp <- VennThemes(C2)
gp[["SetText"]][["Set1"]]$col <-  "red";
gp[["SetText"]][["Set2"]]$cex <- 2;
plot(C2,gp=gp)
# use  highlevel arguments
gp <- VennThemes(C2,colourAlgorithm="binary")
plot(C2,gp=gp)
gp <- VennThemes(C2,increasingLineWidth=TRUE)
plot(C2,gp=gp)
```

Index

∗ **classes**
TissueDrawing-class, 7
Venn-class, 10
VennDrawing-class, 11
∗ **datasets**
StemCell, 6
VennDiagrams, 10
∗ **graphs**
compute.S4, 3
compute.Venn, 4
PlotDarkMatter, 5
PlotVennGeometry, 5
Venn, 9
Vennerable-internal, 12
VennSetSetLabels, 13
VennThemes, 14
∗ **package**
Vennerable-package, 2
.checkPointOnEdge,VDedgeLines-method
(Vennerable-internal), 12
.checkPointOnEdge,VDedgeSector-method
(Vennerable-internal), 12
.checkPointOnEdge-methods
(Vennerable-internal), 12
.edge.to.xy,VDedgeLines,missing-method
(Vennerable-internal), 12
.edge.to.xy,VDedgeLines,numeric-method
(Vennerable-internal), 12
.edge.to.xy,VDedgeSector,missing-method
(Vennerable-internal), 12
.edge.to.xy,VDedgeSector,numeric-method
(Vennerable-internal), 12
.edge.to.xy-methods
(Vennerable-internal), 12
.findIntersectionByType,VDedgeLines,VDedgeLines-method
(Vennerable-internal), 12
.findIntersectionByType,VDedgeLines,VDedgeSector-method
(Vennerable-internal), 12
.findIntersectionByType,VDedgeSector,VDedgeLines-method
(Vennerable-internal), 12
.findIntersectionByType,VDedgeSector,VDedgeSector-method
(Vennerable-internal), 12
.findIntersectionByType-methods
(Vennerable-internal), 12
.identical,VDedgeLines,VDedgeLines-method
(Vennerable-internal), 12
.identical,VDedgeLines,VDedgeSector-method
(Vennerable-internal), 12
.identical,VDedgeSector,VDedgeLines-method
(Vennerable-internal), 12
.identical,VDedgeSector,VDedgeSector-method
(Vennerable-internal), 12
.identical-methods
(Vennerable-internal), 12
.midpoint,VDedgeLines-method
(Vennerable-internal), 12
.midpoint,VDedgeSector-method
(Vennerable-internal), 12
.midpoint-methods
(Vennerable-internal), 12
.reverseEdge,VDedgeLines-method
(Vennerable-internal), 12
.reverseEdge,VDedgeSector-method
(Vennerable-internal), 12
.reverseEdge-methods
(Vennerable-internal), 12
.splitEdgeAtPoint,VDedgeLines-method
(Vennerable-internal), 12
.splitEdgeAtPoint,VDedgeSector-method
(Vennerable-internal), 12
.splitEdgeAtPoint-methods
(Vennerable-internal), 12
[,Venn-method *(Venn-class)*, 10

addCovers *(Vennerable-internal)*, 12
addFace *(Vennerable-internal)*, 12
addSegment *(Vennerable-internal)*, 12
addSetToDrawing *(Vennerable-internal)*, 12

Areas (Vennerable-internal), 12
Areas,VennDrawing-method (VennDrawing-class), 11

battle.function (Vennerable-internal), 12
buildVennDiagrams (Vennerable-internal), 12

CircleDrawing-class (TissueDrawing-class), 7
cog.function (Vennerable-internal), 12
compute.AWFE (Vennerable-internal), 12
compute.C2 (Vennerable-internal), 12
compute.C3 (Vennerable-internal), 12
compute.CR (Vennerable-internal), 12
compute.delta (Vennerable-internal), 12
compute.E4 (Vennerable-internal), 12
compute.S2 (Vennerable-internal), 12
compute.S3 (Vennerable-internal), 12
compute.S4, 3
compute.T3 (Vennerable-internal), 12
compute.Venn, 4
ComputeAreas (Vennerable-internal), 12
CreateViewport (Vennerable-internal), 12
cutAWFE (Vennerable-internal), 12
cutAWFE4 (Vennerable-internal), 12

dark.matter.signature (Vennerable-internal), 12
deleteFace (Vennerable-internal), 12
deltagivenouter (Vennerable-internal), 12
deltasmooth (Vennerable-internal), 12

Euler.from.Signature (Vennerable-internal), 12
EulerClasses (Vennerable-internal), 12

faceAreas (Vennerable-internal), 12
FaceColours (Vennerable-internal), 12
FaceTextColours (Vennerable-internal), 12
fequal (Vennerable-internal), 12

getEdge (Vennerable-internal), 12
getFace (Vennerable-internal), 12

Indicator (Vennerable-internal), 12
injectEdge (Vennerable-internal), 12
injectPoint (Vennerable-internal), 12
injectPoints (Vennerable-internal), 12
internalPointsofFaces (Vennerable-internal), 12
IntersectionMidpoints (Vennerable-internal), 12
IntersectionMidpoints,VennDrawing-method (VennDrawing-class), 11

joinEdges (Vennerable-internal), 12
joinEdges,VDedgeLines,VDedgeLines-method (Vennerable-internal), 12
joinEdges,VDedgeSector,VDedgeSector-method (Vennerable-internal), 12
joinEdges-methods (Vennerable-internal), 12
joinEdgesInDrawing (Vennerable-internal), 12

make.E4 (Vennerable-internal), 12
make.maxiray (Vennerable-internal), 12
make.setlist (Vennerable-internal), 12
makeAWFE (Vennerable-internal), 12
makeAWFE4star (Vennerable-internal), 12
makeAWFESets (Vennerable-internal), 12
makeAWFEstar (Vennerable-internal), 12
makeirs (Vennerable-internal), 12
makePMSGn (Vennerable-internal), 12
makeQn (Vennerable-internal), 12
makeSCD (Vennerable-internal), 12
makesrp (Vennerable-internal), 12
makevp.eqsc (Vennerable-internal), 12
matched.parentheses (Vennerable-internal), 12
my.tsort (Vennerable-internal), 12

newEdgeLines (Vennerable-internal), 12
newEdgeSector (Vennerable-internal), 12
newTissueFromCircle (Vennerable-internal), 12
newTissueFromEllipse (Vennerable-internal), 12
newTissueFromPolygon (Vennerable-internal), 12
node.to.ray (Vennerable-internal), 12
NumberOfSets (Vennerable-internal), 12
NumberOfSets,Venn-method (Venn-class), 10

only.set.crossings
(Vennerable-internal), 12

plot,Venn,missing-method (Venn-class), 10
plot,VennDrawing,missing-method (VennDrawing-class), 11
PlotDarkMatter, 5
PlotFaces (Vennerable-internal), 12
PlotFaces,TissueDrawing-method (TissueDrawing-class), 7
PlotIntersectionText (Vennerable-internal), 12
PlotNodes (Vennerable-internal), 12
PlotNodes,TissueDrawing-method (TissueDrawing-class), 7
PlotSetBoundaries (Vennerable-internal), 12
PlotSetBoundaries,TissueDrawing-method (TissueDrawing-class), 7
PlotSetLabels (Vennerable-internal), 12
PlotUniverse (Vennerable-internal), 12
PlotUniverse,VennDrawing-method (VennDrawing-class), 11
plotVenn (Vennerable-internal), 12
PlotVennGeometry, 5
plotxygraph (Vennerable-internal), 12
pnpoly (Vennerable-internal), 12
pnpolytest (Vennerable-internal), 12
projection.thetaphi (Vennerable-internal), 12

RColorBrewer, *14, 15*
rectifyEdges (Vennerable-internal), 12
remove.nonintersectionpoints (Vennerable-internal), 12
rename.node (Vennerable-internal), 12
renameFaces (Vennerable-internal), 12

scythe.AWFE (Vennerable-internal), 12
scythegr (Vennerable-internal), 12
sector.to.xy (Vennerable-internal), 12
set.function (Vennerable-internal), 12
SetColours (Vennerable-internal), 12
Setfun (Vennerable-internal), 12
SetNames,Venn-method (Venn-class), 10
setSignature (Vennerable-internal), 12
SetTextColours (Vennerable-internal), 12
show,TDFaceList-method (Vennerable-internal), 12
show,TissueDrawing-method (TissueDrawing-class), 7
show,VDedgeDrawn-method (Vennerable-internal), 12
show,VDedgeLines-method (Vennerable-internal), 12
show,VDedgeSector-method (Vennerable-internal), 12
show,Venn-method (Venn-class), 10
show,VennDrawing-method (VennDrawing-class), 11
Smithn.function (Vennerable-internal), 12
spliceEdgeIntoFace (Vennerable-internal), 12
spliceinstead (Vennerable-internal), 12
SquareDrawing-class (TissueDrawing-class), 7
StemCell, 6

TDEdgeList, *8*, *12*
TDEdgeList-class (Vennerable-internal), 12
TDFaceList, *8*, *12*
TDFaceList-class (Vennerable-internal), 12
TDtograph (Vennerable-internal), 12
thetah.to.xy (Vennerable-internal), 12
ThreeCircles (Vennerable-internal), 12
TissueDrawing, *12*
TissueDrawing-class, 7
TwoCircles (Vennerable-internal), 12

UniverseRange (Vennerable-internal), 12
UniverseRange,VennDrawing-method (VennDrawing-class), 11
updateSignature (Vennerable-internal), 12
UpViewports (Vennerable-internal), 12

VDedgeDrawn-class (Vennerable-internal), 12
VDedgeLines-class (Vennerable-internal), 12
VDedgeSector-class (Vennerable-internal), 12
VDPlotArc (Vennerable-internal), 12

Venn, 9, *10*, *12*
Venn-class, 10
VennDiagrams, 10
VennDrawing-class, 11
Vennerable *(*Vennerable-package*)*, 2
Vennerable-internal, 12
Vennerable-package, 2
VennFromSets *(*Vennerable-internal*)*, 12
VennGetFaceLabels *(*VennSetSetLabels*)*, 13
VennGetFaceLabels,VennDrawing-method
*(*VennDrawing-class*)*, 11
VennGetSetLabels *(*VennSetSetLabels*)*, 13
VennGetSetLabels,VennDrawing-method
*(*VennDrawing-class*)*, 11
VennGetUniverseRange
*(*VennSetSetLabels*)*, 13
VennGetUniverseRange,VennDrawing-method
*(*VennSetSetLabels*)*, 13
VennSetFaceLabels *(*VennSetSetLabels*)*, 13
VennSetFaceLabels,VennDrawing-method
*(*VennDrawing-class*)*, 11
VennSetNames *(*Vennerable-internal*)*, 12
VennSetSetLabels, 13
VennSetSetLabels,VennDrawing-method
*(*VennDrawing-class*)*, 11
VennSetSetLabels-methods
*(*VennDrawing-class*)*, 11
VennSetUniverseRange
*(*VennSetSetLabels*)*, 13
VennSetUniverseRange,VennDrawing-method
*(*VennSetSetLabels*)*, 13
VennSignature *(*Vennerable-internal*)*, 12
VennThemes, 14
VisibleRange *(*Vennerable-internal*)*, 12
VisibleRange,TissueDrawing-method
*(*VennDrawing-class*)*, 11
VisibleRange-methods
*(*VennDrawing-class*)*, 11

Weights *(*Venn*)*, 9
Weights<- *(*Venn*)*, 9

zeropos *(*Vennerable-internal*)*, 12
zerotheta *(*Vennerable-internal*)*, 12