

# Package ‘rggobi’

May 10, 2009

**Version** 2.1.14

**Date** 2009-5-8

**Title** Interface between R and GGobi

**Author** Duncan Temple Lang <duncan@research.bell-labs.com>, Debby Swayne  
<dfs@research.att.com>, Hadley Wickham <hadley@iastate.edu>, Michael Lawrence  
<lawremi@iastate.edu>

**Depends** R (>= 2.5.1)

**Suggests** reshape

**Imports** RGtk2, utils, methods

**SystemRequirements** GGobi

**Maintainer** Hadley Wickham <hadley@iastate.edu>

**Description** The rggobi package provides a command-line interface to GGobi, an interactive and dynamic graphics package. Rggobi complements GGobi’s graphical user interface, providing a way to fluidly transition between analysis and exploration, as well as automating common tasks.

**License** BSD

**LazyData** true

**URL** <http://www.ggobi.org/rggobi>

**Acknowledgements** Di Cook, Nicholas Lewin-Koh, Xuejing Chen.

**Repository** CRAN

**Date/Publication** 2009-05-10 19:00:41

**R topics documented:**

"colorscheme<-"	3
"edges<-GGobi"	3
"edges<-GGobiData"	4
"edges<-GGobiDisplay"	5
"excluded<-GGobiData"	6
"glyph_colour<-GGobiData"	6
"glyph_size<-GGobiData"	7
"glyph_type<-GGobiData"	8
"ids<-GGobiData"	8
"shadowed<-GGobiData"	9
"variables<-GGobiDisplay"	9
"[GGobi"	10
"[GGobiData"	11
"[<-GGobi"	12
.check_versions	12
close.GGobi	13
colorscheme	13
connecting_edges	14
dataset.GGobiDisplay	14
display.GGobiData	15
displays.GGobi	16
edges	17
excluded.GGobiData	17
ggobi.default	18
ggobi_count	19
ggobi_display_get_tour_projection	20
ggobi_display_save_picture	21
ggobi_display_set_tour_projection	21
ggobi_get	22
ggobi_longitudinal	23
ggobi_version	24
glyph_colour.GGobiData	24
glyph_size.GGobiData	25
glyph_type.GGobiData	26
ids.GGobiData	27
imode	27
names.GGobi	28
pmode	28
selected.GGobiData	29
shadowed.GGobiData	30
Storm tracks data	30
summary.GGobi	31
summary.GGobiData	32
variables.GGobiDisplay	32

---

"colorscheme<-" *Set active colour scheme.*

---

### Description

Specify the active color scheme in a GGobi instance or the session options.

### Usage

```
"colorscheme<-"(x, value)
```

### Arguments

x	GGobi object
value	colour scheme to make active

### Details

This makes a particular color scheme active within a GGobi instance.

### Value

The name of the previously active color scheme.

### Author(s)

Hadley Wickham <h.wickham@gmail.com>

### Examples

```
g <- ggobi(mtcars)
colorscheme(g) <- "Set1 8"
colorscheme(g) <- 1
```

---

"edges<-.GGobi" *Set edges*

---

### Description

Create a new edges dataset and add to GGobi

### Usage

```
"edges<-.GGobi"(x, value)
```

**Arguments**

`x`  
`value`

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

---

`"edges<-GGobiData"`  
*Set edges*

---

**Description**

Set edges for a dataset.

**Usage**

```
"edges<-GGobiData"(x, value)
```

**Arguments**

<code>x</code>	GGobiData
<code>value</code>	matrix, data frame, or graph containing of edges. First column should be from edge, second column to edge.

**Details**

In GGobi, and edge dataset is a special type of dataset that has two additional (hidden) columns which specification source and destination row names. These rownames are compared to the row names of the dataset in the current plot, and if any match, it is possible to specify this dataset as an edge set to the plotted dataset. When this is done, edges will be drawn between points specified by the edge dataset.

To remove edges, set edges to NULL.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**See Also**

[ggobi\\_longitudinal](#) for creating edges which simulate time series plots

## Examples

```
cc<-cor(t(swiss),use="p", method="s")
ccd<-sqrt(2*(1-cc))
a <- which(lower.tri(ccd), arr.ind=TRUE)
src <- row.names(swiss)[a[,2]]
dest <- row.names(swiss)[a[,1]]
weight <- as.vector(as.dist(ccd))
gg <- ggobi(swiss)
gg$cor <- data.frame(weight)
edges(gg$cor) <- cbind(src, dest)
edges(gg$cor)
edges(gg$cor) <- NULL
```

---

```
"edges<-GGobiDisplay"
  Set edges
```

---

## Description

Set edges for a display

## Usage

```
"edges<-GGobiDisplay"(x, value)
```

## Arguments

x	GGobiDisplay object
value	GGobiData object that contains edges

## Details

This sets the dataset that a GGobiDisplay uses to display edges.

## Author(s)

Hadley Wickham <h.wickham@gmail.com>

---

```
"excluded<-.GGobiData"  
  Set excluded status
```

---

**Description**

Set the exclusion status of points.

**Usage**

```
"excluded<-.GGobiData" (x, value)
```

**Arguments**

x	GGobiData
value	logical vector

**Details**

If a point is excluded it is not drawn.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**See Also**

[excluded](#)

---

```
"glyph_colour<-.GGobiData"  
  Set glyph colour
```

---

**Description**

Set glyph colour

**Usage**

```
"glyph_colour<-.GGobiData" (x, value)
```

**Arguments**

x	GGobiData
value	vector of new colours

"glyph\_size<-.GGobiData"

7

### Author(s)

Hadley Wickham <h.wickham@gmail.com>

### See Also

[glyph\\_colour](#)

---

"glyph\_size<-.GGobiData"  
*Set glyph size*

---

### Description

Set glyph size

### Usage

```
"glyph_size<-.GGobiData"(x, value)
```

### Arguments

x	GGobiData
value	vector of new sizes

### Details

Glyph size is an integer between 1 and 6.

### Author(s)

Hadley Wickham <h.wickham@gmail.com>

### See Also

[glyph\\_size](#)

---

```
"glyph_type<-.GGobiData"  
  Set glyph type
```

---

**Description**

Set glyph type

**Usage**

```
"glyph_type<-.GGobiData"(x, value)
```

**Arguments**

x	GGobiData
value	vector of new types

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**See Also**

[glyph\\_type](#)

---

```
"ids<-.GGobiData"  Set row ids
```

---

**Description**

Set row ids from a GGobiData

**Usage**

```
"ids<-.GGobiData"(x, value)
```

**Arguments**

x	GGobiData
value	new values

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**See Also**

[ids](#)

---

"shadowed<-.GGobiData"  
*Set shadowed status*

---

### Description

Set the exclusion status of points.

### Usage

```
"shadowed<-.GGobiData" (x, value)
```

### Arguments

x	GGobiData
value	logical vector

### Details

If a point is shadowed it is drawn in a dark gray colour, behind all non-shadowed points. It cannot be selected.

### Author(s)

Hadley Wickham <h.wickham@gmail.com>

### See Also

[shadowed](#)

---

"variables<-.GGobiDisplay"  
*Set display variables*

---

### Description

Set display variables with a list of x, y, and z component variable indices.

### Usage

```
"variables<-.GGobiDisplay" (x, value)
```

### Arguments

x	GGobiDisplay object
value	list with X, Y and Z components listing the variable indices to display, either as numeric position or character variable name

**Details**

There are three types of variables in GGobi displays: x, y, z, which correspond to the labels on the toggle buttons in GGobi. Most plots have a constrained set of possible options. For example, in tours you can only set x variables, and you must have at least three. Or in the rotation plot, you need exactly one x, y, and z variable.

Currently, there is no checking done to ensure that you are sending a sensible set of variables for the given display type. Generally, any invalid choices will be silently ignored.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
g <- ggobi(mtcars)
d <- display(g[1], "Parallel Coordinates Display")
variables(d)
variables(d) <- list(X=1:8)
variables(d) <- list(X=c("mpg", "cyl"))
variables(d)
```

---

".GGobi"

*Get ggobi data.*

---

**Description**

Conveniently retrieve ggobi dataset.

**Usage**

```
".GGobi"(x, i, ..., drop=TRUE)
```

**Arguments**

x	GGobi object
i	name of dataset to retrieve
...	
drop	

**Details**

It is convenient to be able to refer to and operate on a ggobi dataset as if it were a regular R dataset. This function allows one to get an `GGobiData` object that represents a particular dataset.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
g <- ggobi(ChickWeight)
g["cars"] <- mtcars
g[1:2]
g["ChickWeight"]
g["cars"]
g$cars
```

---

"[.GGobiData"      *Subsetting*

---

**Description**

Subsetting for ggobi datasets

**Usage**

```
"[.GGobiData"(x, i, j, drop=FALSE)
```

**Arguments**

x	arguments for generic data.frame subset function
i	drop dimensions?
j	
drop	

**Details**

This functions allow one to treat a ggobi dataset as if it were a local data.frame. One can extract and assign elements within the dataset.

This method works by retrieving the entire dataset into R, and then subsetting with R.

**Value**

desired subset from data.frame

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
g <- ggobi(mtcars)
x <- g$mtcars
x[1:5, 1:5]
x[[1]]
x$cyl
```

---

```
"[<-.GGobi"          [<-.GGobi
```

---

**Description**

Add data to ggobi instance.

**Usage**

```
"[<-.GGobi"(x, i, value)
```

**Arguments**

x	ggobi instance
i	name of data frame
value	data.frame, or string to path of file to load

**Details**

This function allows you to add (and eventually) replace GGobiData objects in a GGobi instance.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
g <- ggobi()
g["a"] <- mtcars
g$b <- mtcars
```

---

```
.check_versions      check that rggobi and GGobi by major.minor versions are the same
```

---

**Description**

to ensure binary compatibility

**Usage**

```
.check_versions()
```

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

---

close.GGobi	<i>Close GGobi instance</i>
-------------	-----------------------------

---

**Description**

Terminates and discards a ggobi instance

**Usage**

```
close.GGobi(con, ...)
```

**Arguments**

con	ggobi object to close
...	ignored and for compatability generic function.

**Details**

This allows the caller to close a ggobi instance and discard the resources it uses. The function closes the display windows and variable panel window associated with this ggobi instance. It also resets the default ggobi instance to be the last one created.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
g1 <- ggobi(mtcars)
g2 <- ggobi(mtcars)
close(g2)
close(ggobi_get())
```

---

colorscheme	<i>Get active colour scheme</i>
-------------	---------------------------------

---

**Description**

Get name of the active colour scheme

**Usage**

```
colorscheme(x)
```

**Arguments**

x	GGobi object
---	--------------

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
g <- ggobi(mtcars)
colorscheme(g)
```

---

```
connecting_edges    Get connecting edges
```

---

**Description**

Get actual edges from application of edges dataset to target dataset.

**Usage**

```
connecting_edges(x, y)
```

**Arguments**

x	target ggobi dataset
y	ggobi dataset containing edges

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

---

```
dataset.GGobiDisplay  
    Get display dataset
```

---

**Description**

Returns a link to the GGobiData (dataset) object associated with this display.

**Usage**

```
dataset.GGobiDisplay(x, .gobi=ggobi(x))
```

**Arguments**

x	GGobiDisplay object
.gobi	

## Details

See [\[.GGobi\]](#) for more information on

## Author(s)

Hadley Wickham <h.wickham@gmail.com>

## Examples

```
g <- ggobi(mtcars)
d <- displays(g)[[1]]
dataset(d)
```

---

display.GGobiData *Create a new display*

---

## Description

Create a new display for the GGobiData object.

## Usage

```
display.GGobiData(x, pmode="Scatterplot Display", vars=list(X=names(x)), embed=FALSE)
```

## Arguments

x	projection mode to use
pmode	variables to display, see <a href="#">variables.GGobiDisplay</a> for details
vars	If TRUE, returns widget for use with RGtk2
embed	
...	

## Details

This function allows you to create a new display from a GGobiData object. You will need to specify the type of display you want ("Scatterplot Display", "Scatterplot Matrix" and "Parallel Coordinates Display" are the most common), and which variables the plot should be initialised with. Specifying more than two variables only makes sense for scatterplot matrices and pcps.

Many of the plots used in GGobi (eg. the tours and densities plots) are special modes of the scatterplot display. You will need to create a new scatterplot display, change the projection mode to what you want, and then set the variables. Hopefully this will be improved in a future version of rggobi.

## Author(s)

Hadley Wickham <h.wickham@gmail.com>

**See Also**

[ggobi\\_display\\_types](#) for a list of display types

**Examples**

```
g <- ggobi(mtcars)
display(g[1])
display(g[1], vars=list(X=4, Y=5))
display(g[1], vars=list(X="drat", Y="hp"))
display(g[1], "Parallel Coordinates Display")
## Not run:
display(g[1], "2D Tour")
display(g[1], "2x1D Tour", list(X=c(1,2,3), Y=c(4,5,6)))
## End(Not run)
display(g[1], "Scatterplot Matrix")
```

---

displays.GGobi      *Get GGobi displays*

---

**Description**

Gets list of displays in the specified GGobi instance

**Usage**

```
displays.GGobi(x)
```

**Arguments**

x                    GGobi object

**Details**

A display basically corresponds to a window in GGobi. A display may contain multiple plots within it. For example, the scatterplot matrix contains  $p * p$  plots.

Use this function to obtain a reference to a display (they are numbered in the order they are created) so you can change display mode, set variables ([variables<- .GGobiDisplay](#)), or save a static image to disk.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**See Also**

[display](#) to create displays

**Examples**

```
g <- ggobi(mtcars)
displays(g)
display(g[1])
displays(g)
```

---

edges	<i>Get edges</i>
-------	------------------

---

**Description**

Get edges for a dataset

**Usage**

```
edges(x)
```

**Arguments**

x                    ggobi dataset

**Value**

A matrix of edge pairs

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

---

excluded.GGobiData	<i>Get excluded status</i>
--------------------	----------------------------

---

**Description**

Get the exclusion status of points.

**Usage**

```
excluded.GGobiData(x)
```

**Arguments**

x                    ggobiDataget

**Details**

If a point is excluded it is not drawn.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**See Also**

[excluded<-](#)

---

ggobi.default

*New ggobi*

---

**Description**

Creates a new ggobi instance

**Usage**

```
ggobi.default(data, args=character(0), mode=character(0), name = deparse(sys.call()))
```

**Arguments**

data	the name of a file containing the data, or a data frame or matrix containing the values
args	a character vector of command-line arguments
mode	data format GGobi should expect to read the data from, if reading from a file.
name	the name to use in GGobi for the dataset, if one is specified
...	

**Details**

This function creates a new instance of GGobi with or without new data. Use this function whenever you want to create a new GGobi independent of the others—they will not share linked plots. If you want to add another dataset to an existing ggobi, please see [\[<- .GGobi](#)

There are currently three basic types of functions that you can use with rggobi:

- Data getting and setting: see [\[ .GGobi](#), and [\[ .GGobiData](#)
- "Automatic" brushing: see [glyph\\_colour](#), [glyph\\_size](#), [glyph\\_type](#), [shadowed](#), [excluded](#), and the associated setter functions.
- Edge modification: see [edges](#), [edges<-](#), [ggobi\\_longitudinal](#)

You will generally spend most of your time working with `ggobdats`, you retrieve using `$.GGobiData`, [\[ .GGobiData](#), or [\[\[ .GGobiData](#). Most of the time these will operate like normal R datasets while pointing to the data in GGobi so that all changes are kept in sync. If you need to force a `ggobiDataset` to be an R `data.frame` use [as.data.frame](#).

**Value**

A ggobi object

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
ggobi(ggobi_find_file("data", "flea.csv"))
ggobi(ggobi_find_file("data", "flea.xml"))
ggobi(mtcars)
mtcarsg <- ggobi_get()$mtcars
glyph_colour(mtcarsg)
glyph_colour(mtcarsg) <- ifelse(mtcarsg$cyl < 4, 1, 2)
glyph_size(mtcarsg) <- mtcarsg$cyl
```

---

ggobi\_count

*Get number of GGobis*

---

**Description**

Retrieves the number of ggobi instances within this session

**Usage**

```
ggobi_count()
```

**Details**

One or more ggobi instances can be created within an R session so that one can simultaneously look at different datasets or have different views of the same dataset. This function returns the number of ggobis in existence.

The different ggobi instances are maintained as C level structures. This function accesses a variable that stores how many are in existence when the function is invoked.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
ggobi_count()
```

---

```
ggobi_display_get_tour_projection
  Get tour projection
```

---

## Description

Get the tour projection from a GGobi tour.

## Usage

```
ggobi_display_get_tour_projection(gd)
```

## Arguments

gd

## Details

This function retrieves the current projection matrix from a paused tour. (The tour must be paused so that R can run commands).

This can be used to record interesting projections of your data for later analysis.

## Author(s)

Hadley Wickham <h.wickham@gmail.com>

## Examples

```
g <- ggobi(mtcars)
d <- displays(g)[[1]]
## Not run:
pmode(d) <- "2D Tour"
ggobi_display_get_tour_projection(d)
variables(d) <- list(X=names(mtcars))
ggobi_display_get_tour_projection(d)
MASS::eqsplot(as.matrix(mtcars) %*% ggobi_display_get_tour_projection(d))
## End(Not run)
```

---

```
ggobi_display_save_picture
```

*Save picture of plot (and window) to disk*

---

**Description**

This allows you to make a static copy of a GGobiDisplay.

**Usage**

```
ggobi_display_save_picture(display=displays(ggobi_get())[[1]], path="ggobi_display.
```

**Arguments**

<code>display</code>	GGobiDisplay to save
<code>path</code>	path to save to
<code>filetype</code>	type of file to save
<code>plot.only</code>	if TRUE, save only plot, otherwise save surrounding GUI elements as well

**Details**

If you want to make publication quality graphics, you should probably use the DescribeDisplay plugin and package. This will recreate a GGobiDisplay in R, and so can produce high-quality vector (eg. pdf) output. See <http://www.ggobi.org/describe-display> for more information

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
g <- ggobi(mtcars)
ggobi_display_save_picture(displays(g)[[1]], "test.png")
```

---

```
ggobi_display_set_tour_projection
```

*Set tour projection*

---

**Description**

Set the tour projection from a GGobi tour.

**Usage**

```
ggobi_display_set_tour_projection(gd, value)
```

**Arguments**

gd  
value

**Details**

If you know the projection you would like to see in the tour, you can use this function to set it. The example illustrates setting the projection to show the first two principle components.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
g <- ggobi(mtcars)
d <- displays(g)[[1]]
## Not run:
pmode(d) <- "2D Tour"
variables(d) <- list(X=names(mtcars))
ggobi_display_get_tour_projection(d)
pc <- princomp(as.matrix(mtcars))$loadings[,1:2]
ggobi_display_set_tour_projection(d, pc)
pc <- princomp(as.matrix(mtcars), cor=T)$loadings
ggobi_display_set_tour_projection(d, pc)[,1:2]
## End(Not run)
```

---

ggobi\_get

*Get GGobi*


---

**Description**

Returns a ggobi reference

**Usage**

```
ggobi_get(id = ggobi_count(), drop=TRUE)
```

**Arguments**

id                    numeric vector indicating which ggobi instances to retrieve. Use default if none specified

drop

**Details**

This allows one to get a list of all the ggobi instances currently in existence in the R session. Also, one can fetch particular instances.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
ggobi(mtcars)
ggobi(Nile)
ggobi_get(1)
ggobi_get(1:2)
```

---

ggobi\_longitudinal *Create longitudinal dataset.*

---

**Description**

Instantiate new ggobi with a longitudinal data set.

**Usage**

```
ggobi_longitudinal(data, time=1:rows, id=rep(1, rows), g = NULL)
```

**Arguments**

data	data frame
time	time variable
id	id variable
g	ggobi instance, if you don't want to create a new one

**Details**

This function allows you to load longitudinal data in to GGobi and display it as a line plot. This is achieved by creating edges between adjacent time points, for a given id variable.

For best viewing, we recommend that you turn the show points off in the options menu. When brushing, you may also want to use categorical brushing on the id variable, so that the entire series is selected for an observation.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
data(Oxboys, package="nlme")
ggobi_longitudinal(Oxboys, Occasion, Subject)
ggobi_longitudinal(stormtracks, seasday, id)
ggobi_longitudinal(data.frame(x=1:100, y=sin(1:100)))
```

---

ggobi\_version      *Get version*

---

**Description**

GGobi version information

**Usage**

```
ggobi_version()
```

**Details**

Return an object that describes the version of the ggobi library being used. This allows code to execute conditionally on certain version numbers, etc.

**Value**

date              the release date of the ggobi library  
version           a vector of 3 integers containing the major, minor and patch-level numbers.  
versionstring    a string version of the major, minor and patch-level numbers,

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
ggobi_version()
```

---

glyph\_colour.GGobiData  
*Get glyph colour*

---

**Description**

Get glyph colour

**Usage**

```
glyph_colour.GGobiData(x)
```

**Arguments**

x                  GGobiData

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**See Also**

`glyph_colour<-`

---

`glyph_size.GGobiData`  
*Get glyph size*

---

**Description**

Get glyph size

**Usage**

`glyph_size.GGobiData(x)`

**Arguments**

x                   GGobiData

**Details**

Glyph size is an integer between 1 and 6.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**See Also**

`glyph_size<-`

---

```
glyph_type.GGobiData
```

*Get glyph type.*

---

**Description**

Get glyph type.

**Usage**

```
glyph_type.GGobiData(x)
```

**Arguments**

x                   GGobiData

**Details**

Glyph type refers to the shape of the glyph, one of:

- a filled circle
- an empty circle
- a filled square
- an empty square
- a single pixel
- a plus sign
- a cross

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**See Also**

`glyph_type<-`

---

ids.GGobiData	<i>Row ids</i>
---------------	----------------

---

**Description**

Retrieve row ids from a GGobiData

**Usage**

```
ids.GGobiData(x)
```

**Arguments**

x                   GGobiData

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**See Also**

[ids<-](#)

---

imode	<i>Interaction mode</i>
-------	-------------------------

---

**Description**

Functions to get and set interaction mode

**Usage**

```
imode(x)
```

**Arguments**

x                   GGobiDisplay object

**Details**

In GGobi, the interaction mode determines the how you interact with a plot: brushing, identify etc. Each projection mode also has a default interaction mode that allows you to select variables and control other parameters of the display

You can see the list of available interaction modes using the [imodes](#) function. This accepts either a GGobiDisplay, or the name of the display type.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
g <- ggobi(mtcars)
d <- displays(g)[[1]]
imode(d)
imodes(d)
imode(d) <- "Brush"
```

---

names.GGobi

*GGobi names*

---

**Description**

Get dataset names

**Usage**

```
names.GGobi(x)
```

**Arguments**

x

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
g <- ggobi(mtcars)
names(g)
```

---

pmode

*Projection mode*

---

**Description**

Functions to get and set projection mode

**Usage**

```
pmode(x)
```

**Arguments**

x GGobiDisplay object

**Details**

In GGobi, the projection mode determines the type of plot that is displayed. For example, a 1D ASH plot, or a 2D tour.

You can see the list of available projection modes using the `pmodes` function. This accepts either a GGobiDisplay, or the name of the display type.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
g <- ggobi(mtcars)
d <- displays(g)[[1]]
pmode(d)
pmodes(d)
pmode(d) <- "1D Plot"
```

---

`selected.GGobiData` *Get selection status*

---

**Description**

Returns logical vector indicating if each point is under the brush

**Usage**

```
selected.GGobiData(x)
```

**Arguments**

x GGobiData  
logical vector

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

---

`shadowed.GGobiData` *Get shadowed status*

---

**Description**

Get the exclusion status of points.

**Usage**

```
shadowed.GGobiData(x)
```

**Arguments**

x                    ggobiDataget

**Details**

If a point is shadowed it is drawn in a dark gray colour, behind all non-shadowed points. It cannot be selected.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**See Also**

[shadowed<-](#)

---

`Storm tracks data` *Storm tracks in the Caribbean*

---

**Description**

The data consist of tropical cyclone tracks through the Atlantic Ocean, Caribbean Sea and Gulf of Mexico from 1995 to 2005. Only “named” storms, those which reached tropical storm status or stronger, are included.

The data originated from the National Hurricane Center’s archive of Tropical Cyclone Reports (<http://www.nhc.noaa.gov/pastall.shtml>). From the NHC, the reports "contain comprehensive information on each tropical cyclone, including synoptic history, meteorological statistics, casualties and damages, and the post-analysis best track (six-hourly positions and intensities)."

This dataset is taken from the post-analysis best track information, which are presented in tabular form in the Tropical Cyclone Reports and came in a variety of electronic formats (PDF, HTML and Microsoft Word documents). The best track tables were then copied to text files and parsed into the comma-separated format in which they currently reside.

The variables are as follows:

- Name: Storm Name
- Year: Year of report
- Month: Month of report
- Day: Day of report (day of the month)
- Hour: Hour of report (0, 6, 12 or 18 in UTC time)
- Latitude: Latitude of the storm's center (degrees North)
- Longitude: Longitude of the storm's center (degrees West)
- Pressure: Air pressure at the storm's center (millibars)
- Wind: Storm's maximum sustained wind speed (knots or nautical miles per hour)
- Type: Storm classification (Tropical Depression, Tropical Storm, Hurricane, Extratropical)
- SeasDay: Day of the hurricane season (days since June 1)

The Tropical Cyclone Reports had a variety of storm type designations and there appeared to be no consistent naming convention for cyclones that were not hurricanes, tropical depressions, or tropical storms. Many of these designations have been combined into the "Extratropical" category in this dataset.

This data was put together by Jon Hobbs, a PhD student at Iowa State. Thanks Jon!

### Usage

```
data(stormtracks)
```

### Format

A data frame with 5519 rows and 24 variables

---

summary.GGobi	<i>GGobi summary</i>
---------------	----------------------

---

### Description

Get a description of the global state of the GGobi session.

### Usage

```
summary.GGobi(object, ...)
```

### Arguments

```
object      ggobi object
...
```

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**Examples**

```
g <- ggobi(mtcars)
summary(g)
```

---

```
summary.GGobiData Summarise GGobiData.
```

---

**Description**

Summarise a GGobiData with dimensions, mode and variable names.

**Usage**

```
summary.GGobiData(object, ...)
```

**Arguments**

```
object      GGobiData
...         
```

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

---

```
variables.GGobiDisplay
Get display variables
```

---

**Description**

List the variables used in a given display

**Usage**

```
variables.GGobiDisplay(x)
```

**Arguments**

```
x              GGobiDisplay object
```

**Details**

There are three types of variables in GGobi displays: X, Y, Z, which correspond to the labels on the toggle buttons in GGobi. Most plots have a constrained set of possible options. For example, in tours you can only set X variables, and you must have at least three. Or in the rotation plot, you need exactly one X, Y, and Z variable. You can figure out what these conditions are by using the toggle buttons in GGobi.

**Author(s)**

Hadley Wickham <h.wickham@gmail.com>

**See Also**

`variables<- .GGobiDisplay` for examples

# Index

- \*Topic **attribute**
  - summary.GGobiData, 31
- \*Topic **color**
  - colorscheme, 12
- \*Topic **datasets**
  - Storm tracks data, 29
- \*Topic **dynamic**
  - close.GGobi, 12
  - display.GGobiData, 14
  - displays.GGobi, 15
  - excluded.GGobiData, 16
  - ggobi.default, 17
  - ggobi\_count, 18
  - ggobi\_display\_get\_tour\_projection, 19
  - ggobi\_display\_set\_tour\_projection, 20
  - ggobi\_get, 21
  - ggobi\_longitudinal, 22
  - ggobi\_version, 23
  - glyph\_colour.GGobiData, 23
  - glyph\_size.GGobiData, 24
  - glyph\_type.GGobiData, 25
  - imode, 26
  - names.GGobi, 27
  - pmode, 27
  - selected.GGobiData, 28
  - shadowed.GGobiData, 29
  - summary.GGobi, 30
  - variables.GGobiDisplay, 31
- \*Topic **hplot**
  - ggobi\_display\_save\_picture, 20
- \*Topic **manip**
  - connecting\_edges, 13
  - dataset.GGobiDisplay, 13
  - edges, 16
  - ids.GGobiData, 26
  - .check\_versions, 11
  - [.GGobi, 14, 17
  - [.GGobiData, 17
  - [<-.GGobi, 17
  - [ [.GGobiData, 17
  - \$.GGobiData, 17
  - as.data.frame, 17
  - close.GGobi, 12
  - colorscheme, 12
  - connecting\_edges, 13
  - dataset.GGobiDisplay, 13
  - display, 15
  - display.GGobiData, 14
  - displays.GGobi, 15
  - edges, 16, 17
  - edges<-, 17
  - excluded, 5, 17
  - excluded.GGobiData, 16
  - excluded<-, 17
  - ggobi (ggobi.default), 17
  - ggobi.default, 17
  - ggobi\_count, 18
  - ggobi\_display\_get\_tour\_projection, 19
  - ggobi\_display\_save\_picture, 20
  - ggobi\_display\_set\_tour\_projection, 20
  - ggobi\_display\_types, 15
  - ggobi\_get, 21
  - ggobi\_longitudinal, 3, 17, 22
  - ggobi\_version, 23
  - glyph\_color
    - (glyph\_colour.GGobiData), 23
  - glyph\_colour, 6, 17

glyph\_colour  
    (*glyph\_colour.GGobiData*),  
    23  
glyph\_colour.GGobiData, 23  
glyph\_colour<-, 24  
glyph\_size, 6, 17  
glyph\_size  
    (*glyph\_size.GGobiData*), 24  
glyph\_size.GGobiData, 24  
glyph\_size<-, 24  
glyph\_type, 7, 17  
glyph\_type  
    (*glyph\_type.GGobiData*), 25  
glyph\_type.GGobiData, 25  
glyph\_type<-, 25

ids, 7  
ids (*ids.GGobiData*), 26  
ids.GGobiData, 26  
ids<-, 26  
imode, 26  
imode<- (*imode*), 26  
imodes, 26

names.GGobi, 27

pmode, 27  
pmode<- (*pmode*), 27  
pmodes, 28

rggobi (*ggobi.default*), 17

selected (*selected.GGobiData*), 28  
selected.GGobiData, 28  
shadowed, 8, 17  
shadowed (*shadowed.GGobiData*), 29  
shadowed.GGobiData, 29  
shadowed<-, 29  
Storm tracks data, 29  
stormtracks (*Storm tracks data*),  
    29  
summary.GGobi, 30  
summary.GGobiData, 31

variables.GGobiDisplay, 14, 31  
variables<- .GGobiDisplay, 15, 32