

# Package ‘DataViz’

January 20, 2025

**Type** Package

**Title** Data Visualisation Using an HTML Page and 'D3.js'

**Version** 0.2.8

**Date** 2019-09-16

**Maintainer** Timothy Bell <horia.yeb@gmail.com>

**Description** Gives access to data visualisation methods that are relevant from the statistician's point of view. Using 'D3's existing data visualisation tools to empower R language and environment. The throw chart method is a line chart used to illustrate paired data sets (such as before-after, male-female).

**License** GPL (>= 2)

**Imports** Rcpp (>= 1.0.0)

**LinkingTo** Rcpp

**Depends** tibble

**LazyData** true

**NeedsCompilation** yes

**Author** Timothy Bell [aut, cre],  
Christophe Genolini [aut, ths]

**Repository** CRAN

**Date/Publication** 2019-09-16 17:10:03 UTC

## Contents

DataViz-package . . . . .	2
forcelayout . . . . .	3
oldestpeople . . . . .	3
quad . . . . .	4
rcpp_forcelayout . . . . .	4
rcpp_throwchart . . . . .	5
r_forcelayout . . . . .	6
r_throwchart . . . . .	6
throwchart . . . . .	7
weekschedule . . . . .	8
Workweek . . . . .	8

---

DataViz-package      ~ Overview: package DataViz ~

---

## Description

Various data vizualisation methods.

## Details

Data Visualisation is the art of graphically representing data. There are numerous data visualisation methods, but they aren't always relevant -and sometimes less informative than basic representations-. Moreover they are often created by programmers in various computer languages and the code being seldom available.

DataViz is a package aiming to give access to Data Visualisation methods that are relevant from the statistician's point of view.

The 3 first methods to be implemented are [throwchart](#), Gravity Bubble Chart (V0.3, june 2019) and XXX (V0.4, septembre 2019)

## Author(s)

Timothy Bell <[horia.yeb@gmail.com](mailto:horia.yeb@gmail.com)> Christophe Genolini <[christophe.genolini@u-paris10.fr](mailto:christophe.genolini@u-paris10.fr)>

Maintainer: Timothy Bell <[horia.yeb@gmail.com](mailto:horia.yeb@gmail.com)>

## References

Inspired from <http://tiffanyfrance.com/data-is-beautiful/19-01/>

## See Also

[throwchart forcelayout](#)

## Examples

```
if(interactive()){
  throwchart(c(1,2),c(2,8),c("#000", "#F82"),id = c("id1","id2"),c(1,5))
  throwchart(c(1,2),c(2,8))
}
if(!interactive()){
  throwchart(c(1,2),c(2,8), offSet = 1, webinteract=FALSE)
  throwchart(c(1,2),c(2,1), webinteract=FALSE)
  throwchart(c(1,2),c(2,8),c("#000", "#F00"),c(1,5), webinteract=FALSE)
}
```

---

forcelayout                    *~ Main function: forcelayout ~*

---

### Description

forcelayout method is a dynamic method showing longitudinal data set evolution.

### Usage

```
forcelayout(schedule, webinteract, ttime)
```

### Arguments

schedule            [numeric] or [integer]: A (non-empty) vector of data values.  
webinteract        [bool]: Is the function used in interactive mode?  
ttime                [string]: A (non-empty) time-unit value to fetch data from data.frame.

### Examples

```
if(interactive()){  
  forcelayout(weekschedule)  
}  
if(!interactive()){  
  forcelayout(weekschedule, webinteract = FALSE, ttime = "Monday")  
}
```

---

oldestpeople                    *~ List of the oldest people in the world data set ~*

---

### Description

This is data from <http://www.grg.org/Adams/Deaths2012.HTM>, a list of all the oldest people in history holding the record.

---

quad ~ *Quadratic fitting function: quad* ~

---

### Description

Fits a set of paired points with a quadratic curve. Returns the quadratic set of points. Function only called by `r_throwchart`.

### Usage

```
quad(point,before_point,after_point,offSet)
```

### Arguments

point	[numeric] or [integer]: A set of points between before and after points.
before_point	[numeric] or [integer]: The first point of the curve.
after_point	[numeric] or [integer]: The last point of the curve.
offSet	[integer]: Single value offset for the graph.

### Value

returns the quadratic point equivalent.

---

rcpp\_forcelayout ~ *C++ called function: rcpp\_forcelayout* ~

---

### Description

This function takes the inputs from [forcelayout](#), and writes the data in Json array, then this function calls a windows cmd function to open an index.html in the extdata.

### Usage

```
rcpp_forcelayout(schedule, path)
```

### Arguments

schedule	A number column
path	The path for the library

### Examples

```
if(interactive()){
  rcpp_throwchart(weekschedule,path.package("DataViz"))
}
```

---

rcpp\_throwchart      ~ C++ called function: rcpp\_throwchart ~

---

## Description

This function takes the inputs from [throwchart](#), and writes the data in Json array, then this function calls a windows cmd function to open an index.html in the extdata.

## Usage

```
rcpp_throwchart(before, after ,col, id, lwd, xlim, ylim, offSet, path)
```

## Arguments

before	A number column
after	A number column
col	A hex code colour colum has to be format "#000"
id	An id has to be string
lwd	A number for the line width best between 1-5)
xlim	[numeric]: 2 value colum with x limits.
ylim	[numeric]: 2 value colum with y limits.
offSet	[integer]: Single value for the graph offset.
path	The path for the library

## Examples

```
if(interactive()){  
  before = tibble(c(1,2))  
  after = tibble(c(2,8))  
  col = tibble(c("#000", "#F82"))  
  id = tibble(c("", ""))  
  lwd = tibble(c(1,5))  
  xlim = tibble(c(0,0))  
  rcpp_throwchart(before,after,col,id,lwd,xlim,0,path.package("DataViz"))}
```

---

r\_forcelayout                    *~ R graphics function: r\_forcelayout ~*

---

### Description

Used when interactive is false and creates a plot through R of this data visualisation method.

### Usage

```
r_forcelayout(schedule, ttime)
```

### Arguments

schedule                    [string]: A (non-empty) data.frame of data values.  
 ttime                        [string]: A (non-empty) time-unit value to fetch data from data.frame.

### Examples

```
if(interactive()){
  r_forcelayout(weekschedule,ttime = "Tuesday")
}
```

---

r\_throwchart                    *~ R graphics function: r\_throwchart ~*

---

### Description

Used when interactive is false and creates a plot through R of this data visualisation method.

### Usage

```
r_throwchart(before, after,xlim, ylim, col, lwd, offSet)
```

### Arguments

before                        [numeric] or [integer]: A (non-empty) vector of data values.  
 after                         [numeric] or [integer]: A (non-empty) vector of data values.  
 col                            [character]: A vector of hex code colours, by default "#123".  
 lwd                            [integer]: Line width, a column of line widths, by default value is 2.5.  
 xlim                          [numeric]: 2 value colum with x limits.  
 ylim                          [numeric]: 2 value colum with y limits.  
 offSet                        [integer]: Single value offset for the graph.

**Examples**

```
if(interactive()){
  r_throwchart(tibble(c(1,2)),tibble(c(2,8)),c(0,0),c(0,0),col = "blue", c(1,5), offSet = 1)
}
```

---

 throwchart

 ~ Main function: throwchart ~
 

---

**Description**

Throwchart method is useful for visualising paired data, such as before/after data sets. Each pair of points are set on a horizontal axis and joined by a parabola. The height of the parabola is proportional to the difference: after- before = difference. If the difference is negative (after < before) then the curve is drawn under the axis.

**Usage**

```
throwchart(before, after, col, id, lwd, xlim, ylim, offSet, webinteract)
```

**Arguments**

before	[numeric] or [integer]: A (non-empty) vector of data values.
after	[numeric] or [integer]: A (non-empty) vector of data values.
col	[character]: A vector of hex code colours, by default "#123".
id	[factor]: Column of string or number identifiers.
lwd	[integer]: Line width, a column of line widths, by default value is 2.5.
xlim	[numeric]: 2 value column with x limits.
ylim	[numeric]: 2 value column with y limits.
offSet	[integer]: Single value offset for the graph.
webinteract	[bool]: Is the function used in interactive mode?

**Examples**

```
if(interactive()){
  throwchart(c(1,2),c(2,8),c("#000","#F82"),id = c("id1","id2"),c(1,5))
  throwchart(c(1,2),c(2,8), offSet = 1, webinteract=TRUE)
}
if(!interactive()){
  throwchart(c(1,2),c(2,8), offSet = 1, webinteract=FALSE)
  throwchart(c(1,2),c(2,1), webinteract=FALSE)
  throwchart(c(1,2),c(2,8),c("#000","#F00"),id = c("id1","id2"),c(1,5), webinteract=FALSE)
}
n <- 10
Avant <- rnorm(n)
Apres <- Avant + rnorm(n) + 10
throwchart(Avant, Apres, xlim = c(-4,14), webinteract = FALSE)
throwchart(Avant, Apres, offSet = 0, webinteract = FALSE)
throwchart(Avant, Apres, offSet = 8, webinteract = FALSE)
}
```

---

weekschedule                      ~ *Example data set number 2 for force layout* ~

---

**Description**

Set for force layout, artificial data.

---

Workweek                              ~ *Example data set for force layout* ~

---

**Description**

Set for force layout, artificial data.



# Index

## \* **DataViz**

DataViz-package, [2](#)

DataViz (DataViz-package), [2](#)

DataViz-package, [2](#)

forcelayout, [2](#), [3](#), [4](#)

oldestpeople, [3](#)

quad, [4](#)

r\_forcelayout, [6](#)

r\_throwchart, [6](#)

rcpp\_forcelayout, [4](#)

rcpp\_throwchart, [5](#)

throwchart, [2](#), [5](#), [7](#)

weekschedule, [8](#)

Workweek, [8](#)