Package 'metapost'

June 25, 2019

Type Package

Title Interface to 'MetaPost'

Version 1.0-6

Author Paul Murrell

Maintainer Paul Murrell <paul@stat.auckland.ac.nz>

Description Provides an interface to 'MetaPost' (Hobby, 1998) <http://www.tug.org/docs/metapost/mpman.pdf>. There are functions to generate an R description of a 'MetaPost' curve, functions to generate 'MetaPost' code from an R description, functions to process 'MetaPost' code, and functions to read solved 'MetaPost' paths back into R.

Imports grid, gridBezier

Suggests grImport

SystemRequirements mpost

URL https://github.com/pmur002/metapost, https://stattech.wordpress.fos.auckland.ac.nz/2018/12/03/2018-12-metapost-three-ways/

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2019-06-24 22:20:03 UTC

R topics documented:

metapost-package *R Interface to MetaPost*

Description

Provides an R interface to METAPOST. There are functions to generate an R description of a MetaPost curve, functions to generate MetaPost code from an R description, functions to process MetaPost code, and functions to read solved MetaPost paths back into R.

Details

Generate a MetaPost path with functions like knot.

Write the MetaPost path to a file with metapost.

Run mpost on the file with mpost.

Read the solved path (Bezier control points) into R with mptrace.

Draw a solved path with grid.metapost.

It is also possible to pass grid.metapost the original path (and it will perform the write/solve/read steps itself).

Author(s)

Paul Murrell <paul@stat.auckland.ac.nz>

References

Hobby, J. D. and the MetaPost development team (2018). METAPOST a user's manual. https://www.tug.org/docs/metapost/mpman.pdf

See Also

knot metapost mpost mptrace grid.metapost

Examples

```
oldunits <- options(metapost.units="in")
p <- knot(0, 0) + dir(0) + dir(0) + knot(1, 1)
grid.metapost(p)
options(oldunits)
```

grid.metapost *Draw a MetaPost curve.*

Description

Draw a MetaPost curve in **grid** graphics.

Usage

```
## S3 method for class 'mppath'
metapostGrob(x, gp = gpar(), name = NULL, digits=2, ...)
## S3 method for class 'mpcontrols'
metapostGrob(x, gp = gpar(), name = NULL, ...)
## S3 method for class 'mpcontrolList'
metapostGrob(x, gp = gpar(), name = NULL, ...)
grid.metapost(...)
```

Arguments

x	A MetaPost path, either unsolved (a description generated using knot etc), or solved (as produced by mptrace).
gp	Graphical parameters (from a call to gpar).
name	A name for the grob that is created.
digits	The number of decimal places to use when writing floating point values in MetaPost code.
...	Arguments passed to metapostGrob.

Value

metapostGrob creates a "metapostgrob" object.

Author(s)

Paul Murrell

See Also

knot, mptrace.

Examples

```
oldunits <- options(metapost.units="in")
p <- knot(0, 0) + dir(0) + dir(0) + knot(1, 1)
grid.metapost(p)
options(oldunits)
```

knot *Create a MetaPost Path*

Description

These functions can be used to describe a MetaPost path, consisting of two or more knots, with various constraints on how the path behaves between the knots.

Usage

```
knot(x, y, units = getOption("metapost.units"),
     dir = NA, dir.left = dir, dir.right = dir,
     cp.left.x = NA, cp.right.x = NA, cp.left.y = NA, cp.right.y = NA,
     curl.left = NA, curl.right = NA,
     tension.left = NA, tension.right = NA)
cp(x, y, units = getOption("metapost.units"))
curl(x)
cycle()
dir(x, y = NULL)
tension(x)
```

Arguments

`x`	Numeric value: a location (for `knot` and `cp`), or an angle (for `dir`, if `y` is `NULL`), or a vector component (for `dir`), or a curl or tension value. Or a **grid** unit (for `knot` and `cp`).
`y`	Numeric value: a location (for `knot` and `cp`), or a vector component (for `dir`). Or a **grid** unit (for `knot` and `cp`).
`units`	The **grid** coordinate system to use for locations (if locations are only given as numeric values).
`dir, dir.left, dir.right`	A numeric angle.
`cp.left.x, cp.right.x, cp.left.y, cp.right.y`	A numeric location.
`curl.left, curl.right`	A numeric curl value (must be at least 0).
`tension.left, tension.right`	A numeric tension value (must be at least 3/4). A negative values indicates a lower bound.

Details

A MetaPost path is constructed using calls to `knot` and combining the results using the `+` operator (see the examples below).

Constraints for a knot can be specified within the call to `knot` or by combining connectors (`cp`, `dir`, etc) with a knot using `+`.

Knots can also be combined using - (a straight line rather than a curve), %+% (no inflection), and %-% (straight line with smooth connection) operators.

Value

The individual functions generate knots and connectors, but when combined together, they produce a MetaPost path ("mppath") object.

Author(s)

Paul Murrell

References

Hobby, J. D. and the MetaPost development team (2018). METAPOST a user's manual. https://www.tug.org/docs/metapost/mpman.pdf

See Also

metapost, mpost, mptrace, grid.metapost.

Examples

```
knot(0, 0, dir.right=0)
knot(0, 0, dir.right=0) + knot(1, 1)
knot(0, 0) + dir(0) + knot(1, 1)
```

---

metapost *Generate a MetaPost File*

Description

Generate a MetaPost file from a MetaPost path.

Usage

```
metapost(x, file = "fig.mp", digits=2)
```

Arguments

x	A MetaPost path, as produced from knot etc.
file	The name of the file to produce. If NULL, no file is created.
digits	The number of decimal places to use when writing floating point values in MetaPost code.

Value

The MetaPost code is returned invisibly.

Author(s)

Paul Murrell

See Also

knot

Examples

```
p <- knot(0, 0) + dir(0) + dir(0) + knot(1, 1)
mpcode <- metapost(p, NULL)
mpcode
```

---

mpost *Run* mpost *on a MetaPost File*

Description

Run mpost on a MetaPost file, possibly with additional options.

Usage

```
mpost(file = "fig.mp",
      cmd = NULL, template = NULL, format = NULL, tracing = TRUE)
```

Arguments

file	The name of a file containing MetaPost code.
cmd	The command to use to run mpost. By default Sys.which is used to find a sensible value.
template	The naming template for the output files that are produced (see mpost's outputtemplate option).
format	The output format (see mpost's outputformat option).
tracing	A logical value indicating whether to generate a log file containing solved paths (see mpost's tracingchoices option).

Details

By default, tracing is TRUE, which produces a log file that can be read into R using mptrace.

Both output and log files will be produced in the same directory as the file.

Value

Used for its side effect of producing output files and log files.

Author(s)

Paul Murrell

See Also

metapost, mptrace

Examples

```
oldunits <- options(metapost.units="in")
p <- knot(0, 0) + dir(0) + dir(0) + knot(1, 1)
mpfile <- file.path(tempdir(), "fig.mp")
metapost(p, mpfile)
mpost(mpfile)
options(oldunits)
```

---

mptrace *Parse* mpost *Log Files*

Description

Read a log file generated by mpost (possibly via mpost) to obtain solved MetaPost path information (which can then be drawn by grid.metapost).

Usage

```
mptrace(logfile = "fig.log")
mpbbox(psfile)
mpvp(psfile, ...)
```

Arguments

logfile	The name of a log file generated by mpost.
psfile	The name of a PostScript files generated by mpost.
...	Arguments passed on to viewport.

Details

The log file must have been generated by mpost with tracingchoices=1 (possibly using mpost(..., tracing=TRUE)).

The functions mpbbox and mpvp parse a PostScript file that was generated by mpost, returning the bounding box of the output and a viewport based on that bounding box respectively.

Value

A list of Bezier control points (mpcontrols objects).

Author(s)

Paul Murrell

See Also

mpost, grid.metapost

Examples

```
oldunits <- options(metapost.units="in")
oldwd <- setwd(tempdir())
p <- knot(0, 0) + dir(0) + dir(0) + knot(1, 1)
metapost(p, "fig.mp")
mpost("fig.mp")
paths <- mptrace("fig.log")
grid.metapost(paths)
setwd(oldwd)
options(oldunits)
```

Index