|  |  |
| --- | --- |
| plotmath {grDevices} | R Documentation |

Mathematical Annotation in R

**Description**

If the text argument to one of the text-drawing functions ([text](http://127.0.0.1:33641/help/library/grDevices/help/text), [mtext](http://127.0.0.1:33641/help/library/grDevices/help/mtext), [axis](http://127.0.0.1:33641/help/library/grDevices/help/axis), [legend](http://127.0.0.1:33641/help/library/grDevices/help/legend)) in**R** is an expression, the argument is interpreted as a mathematical expression and the output will be formatted according to TeX-like rules. Expressions can also be used for titles, subtitles and x- and y-axis labels (but not for axis labels on persp plots).

In most cases other language objects (names and calls, including formulas) are coerced to expressions and so can also be used.

**Details**

A mathematical expression must obey the normal rules of syntax for any **R** expression, but it is interpreted according to very different rules than for normal **R** expressions.

It is possible to produce many different mathematical symbols, generate sub- or superscripts, produce fractions, etc.

The output from demo(plotmath) includes several tables which show the available features. In these tables, the columns of grey text show sample **R** expressions, and the columns of black text show the resulting output.

The available features are also described in the tables below:

|  |  |
| --- | --- |
| **Syntax** | **Meaning** |
| x + y | x plus y |
| x - y | x minus y |
| x\*y | juxtapose x and y |
| x/y | x forwardslash y |
| x %+-% y | x plus or minus y |
| x %/% y | x divided by y |
| x %\*% y | x times y |
| x %.% y | x cdot y |
| x[i] | x subscript i |
| x^2 | x superscript 2 |
| paste(x, y, z) | juxtapose x, y, and z |
| sqrt(x) | square root of x |
| sqrt(x, y) | yth root of x |
| x == y | x equals y |
| x != y | x is not equal to y |
| x < y | x is less than y |
| x <= y | x is less than or equal to y |
| x > y | x is greater than y |
| x >= y | x is greater than or equal to y |
| x %~~% y | x is approximately equal to y |
| x %=~% y | x and y are congruent |
| x %==% y | x is defined as y |
| x %prop% y | x is proportional to y |
| x %~% y | x is distributed as y |
| plain(x) | draw x in normal font |
| bold(x) | draw x in bold font |
| italic(x) | draw x in italic font |
| bolditalic(x) | draw x in bolditalic font |
| symbol(x) | draw x in symbol font |
| list(x, y, z) | comma-separated list |
| ... | ellipsis (height varies) |
| cdots | ellipsis (vertically centred) |
| ldots | ellipsis (at baseline) |
| x %subset% y | x is a proper subset of y |
| x %subseteq% y | x is a subset of y |
| x %notsubset% y | x is not a subset of y |
| x %supset% y | x is a proper superset of y |
| x %supseteq% y | x is a superset of y |
| x %in% y | x is an element of y |
| x %notin% y | x is not an element of y |
| hat(x) | x with a circumflex |
| tilde(x) | x with a tilde |
| dot(x) | x with a dot |
| ring(x) | x with a ring |
| bar(xy) | xy with bar |
| widehat(xy) | xy with a wide circumflex |
| widetilde(xy) | xy with a wide tilde |
| x %<->% y | x double-arrow y |
| x %->% y | x right-arrow y |
| x %<-% y | x left-arrow y |
| x %up% y | x up-arrow y |
| x %down% y | x down-arrow y |
| x %<=>% y | x is equivalent to y |
| x %=>% y | x implies y |
| x %<=% y | y implies x |
| x %dblup% y | x double-up-arrow y |
| x %dbldown% y | x double-down-arrow y |
| alpha -- omega | Greek symbols |
| Alpha -- Omega | uppercase Greek symbols |
| theta1, phi1, sigma1, omega1 | cursive Greek symbols |
| Upsilon1 | capital upsilon with hook |
| aleph | first letter of Hebrew alphabet |
| infinity | infinity symbol |
| partialdiff | partial differential symbol |
| nabla | nabla, gradient symbol |
| 32\*degree | 32 degrees |
| 60\*minute | 60 minutes of angle |
| 30\*second | 30 seconds of angle |
| displaystyle(x) | draw x in normal size (extra spacing) |
| textstyle(x) | draw x in normal size |
| scriptstyle(x) | draw x in small size |
| scriptscriptstyle(x) | draw x in very small size |
| underline(x) | draw x underlined |
| x ~~ y | put extra space between x and y |
| x + phantom(0) + y | leave gap for "0", but don't draw it |
| x + over(1, phantom(0)) | leave vertical gap for "0" (don't draw) |
| frac(x, y) | x over y |
| over(x, y) | x over y |
| atop(x, y) | x over y (no horizontal bar) |
| sum(x[i], i==1, n) | sum x[i] for i equals 1 to n |
| prod(plain(P)(X==x), x) | product of P(X=x) for all values of x |
| integral(f(x)\*dx, a, b) | definite integral of f(x) wrt x |
| union(A[i], i==1, n) | union of A[i] for i equals 1 to n |
| intersect(A[i], i==1, n) | intersection of A[i] |
| lim(f(x), x %->% 0) | limit of f(x) as x tends to 0 |
| min(g(x), x > 0) | minimum of g(x) for x greater than 0 |
| inf(S) | infimum of S |
| sup(S) | supremum of S |
| x^y + z | normal operator precedence |
| x^(y + z) | visible grouping of operands |
| x^{y + z} | invisible grouping of operands |
| group("(",list(a, b),"]") | specify left and right delimiters |
| bgroup("(",atop(x,y),")") | use scalable delimiters |
| group(lceil, x, rceil) | special delimiters |
| group(lfloor, x, rfloor) | special delimiters |
|  |  |

The supported ‘scalable delimiters’ are | ( [ {, lceil, lfloor and their right-hand versions. "." is equivalent to "": the corresponding delimiter will be omitted. Delimiter || is supported but has the same effect as |.

The symbol font uses Adobe Symbol encoding so, for example, a lower case mu can be obtained either by the special symbol mu or by symbol("m"). This provides access to symbols that have no special symbol name, for example, the universal, or forall, symbol issymbol("\042"). To see what symbols are available in this way useTestChars(font=5) as given in the examples for [points](http://127.0.0.1:33641/help/library/grDevices/help/points): some are only available on some devices.

Note to TeX users: TeX's \Upsilon is Upsilon1, TeX's \varepsilon is close toepsilon, and there is no equivalent of TeX's \epsilon. TeX's \varpi is close to omega1.vartheta, varphi and varsigma are allowed as synonyms for theta1, phi1 andsigma1.

sigma1 is also known as stigma, its Unicode name.

Control characters (e.g., \n) are not interpreted in character strings in plotmath, unlike normal plotting.

The fonts used are taken from the current font family, and so can be set by [par](http://127.0.0.1:33641/help/library/grDevices/help/par)(family=)in base graphics, and [gpar](http://127.0.0.1:33641/help/library/grDevices/help/gpar)(fontfamily=) in package **grid**.

Note that bold, italic and bolditalic do not apply to symbols, and hence not to the Greek *symbols* such as mu which are displayed in the symbol font. They also do not apply to numeric constants.

**Other symbols**

On many OSes and some graphics devices many other symbols are available as part of the standard text font, and all of the symbols in the Adobe Symbol encoding are in principle available *via* changing the font face or (see ‘Details’) plotmath: see the examples section of[points](http://127.0.0.1:33641/help/library/grDevices/help/points) for a function to display them. (‘In principle’ because some of the glyphs are missing from some implementations of the symbol font.) Unfortunately, [postscript](http://127.0.0.1:33641/help/library/grDevices/help/postscript) and [pdf](http://127.0.0.1:33641/help/library/grDevices/help/pdf) have support for little more than European (not Greek) and CJK characters and the Adobe Symbol encoding (and in a few fonts, also Cyrillic characters).

Any Unicode character can be entered into a text string *via* a \uxxxx escape, or used by number in a call to [points](http://127.0.0.1:33641/help/library/grDevices/help/points). The [windows](http://127.0.0.1:33641/help/library/grDevices/help/windows) family of devices can display such characters if they are available in the font in use. This can often be used to display Greek *letters* in bold or italic.

A good way to both find out which characters are available in a font and to determine the Unicode number is to use the ‘Character Map’ accessory (usually on the ‘Start’ menu under ‘Accessories->System Tools’). You can also copy-and-paste characters from the ‘Character Map’ window to the Rgui console (but not to Rterm).

**References**

Murrell, P. and Ihaka, R. (2000) An approach to providing mathematical annotation in plots.*Journal of Computational and Graphical Statistics*, **9**, 582–599.

The symbol codes can be found in octal in the Adobe reference manuals, e.g. for Postscript<https://www.adobe.com/products/postscript/pdfs/PLRM.pdf> or PDF<https://www.adobe.com/devnet/acrobat/pdfs/pdf_reference_1-7.pdf> and in decimal, octal and hex at <http://www.stat.auckland.ac.nz/~paul/R/CM/AdobeSym.html>.

**See Also**

demo(plotmath), [axis](http://127.0.0.1:33641/help/library/grDevices/help/axis), [mtext](http://127.0.0.1:33641/help/library/grDevices/help/mtext), [text](http://127.0.0.1:33641/help/library/grDevices/help/text), [title](http://127.0.0.1:33641/help/library/grDevices/help/title), [substitute](http://127.0.0.1:33641/help/library/grDevices/help/substitute) [quote](http://127.0.0.1:33641/help/library/grDevices/help/quote), [bquote](http://127.0.0.1:33641/help/library/grDevices/help/bquote)

**Examples**

require(graphics)

x <- seq(-4, 4, len = 101)

y <- cbind(sin(x), cos(x))

matplot(x, y, type = "l", xaxt = "n",

main = expression(paste(plain(sin) \* phi, " and ",

plain(cos) \* phi)),

ylab = expression("sin" \* phi, "cos" \* phi), # only 1st is taken

xlab = expression(paste("Phase Angle ", phi)),

col.main = "blue")

axis(1, at = c(-pi, -pi/2, 0, pi/2, pi),

labels = expression(-pi, -pi/2, 0, pi/2, pi))

## How to combine "math" and numeric variables :

plot(1:10, type="n", xlab="", ylab="", main = "plot math & numbers")

theta <- 1.23 ; mtext(bquote(hat(theta) == .(theta)), line= .25)

for(i in 2:9)

text(i, i+1, substitute(list(xi, eta) == group("(",list(x,y),")"),

list(x = i, y = i+1)))

## note that both of these use calls rather than expressions.

##

text(1, 10, "Derivatives:", adj = 0)

text(1, 9.6, expression(

" first: {f \* minute}(x) " == {f \* minute}(x)), adj = 0)

text(1, 9.0, expression(

" second: {f \* second}(x) " == {f \* second}(x)), adj = 0)

plot(1:10, 1:10)

text(4, 9, expression(hat(beta) == (X^t \* X)^{-1} \* X^t \* y))

text(4, 8.4, "expression(hat(beta) == (X^t \* X)^{-1} \* X^t \* y)",

cex = .8)

text(4, 7, expression(bar(x) == sum(frac(x[i], n), i==1, n)))

text(4, 6.4, "expression(bar(x) == sum(frac(x[i], n), i==1, n))",

cex = .8)

text(8, 5, expression(paste(frac(1, sigma\*sqrt(2\*pi)), " ",

plain(e)^{frac(-(x-mu)^2, 2\*sigma^2)})),

cex = 1.2)

## some other useful symbols

plot.new(); plot.window(c(0,4), c(15,1))

text(1, 1, "universal", adj = 0); text(2.5, 1, "\\042")

text(3, 1, expression(symbol("\042")))

text(1, 2, "existential", adj = 0); text(2.5, 2, "\\044")

text(3, 2, expression(symbol("\044")))

text(1, 3, "suchthat", adj = 0); text(2.5, 3, "\\047")

text(3, 3, expression(symbol("\047")))

text(1, 4, "therefore", adj = 0); text(2.5, 4, "\\134")

text(3, 4, expression(symbol("\134")))

text(1, 5, "perpendicular", adj = 0); text(2.5, 5, "\\136")

text(3, 5, expression(symbol("\136")))

text(1, 6, "circlemultiply", adj = 0); text(2.5, 6, "\\304")

text(3, 6, expression(symbol("\304")))

text(1, 7, "circleplus", adj = 0); text(2.5, 7, "\\305")

text(3, 7, expression(symbol("\305")))

text(1, 8, "emptyset", adj = 0); text(2.5, 8, "\\306")

text(3, 8, expression(symbol("\306")))

text(1, 9, "angle", adj = 0); text(2.5, 9, "\\320")

text(3, 9, expression(symbol("\320")))

text(1, 10, "leftangle", adj = 0); text(2.5, 10, "\\341")

text(3, 10, expression(symbol("\341")))

text(1, 11, "rightangle", adj = 0); text(2.5, 11, "\\361")

text(3, 11, expression(symbol("\361")))

[Package *grDevices* version 3.3.0 [Index](http://127.0.0.1:33641/help/library/grDevices/html/00Index.html)]