

Package ‘Ryacas’

April 17, 2009

Version 0.2-9

Date 2008-12-19

Title R interface to the yacas computer algebra system

Author Rob Goedman <goedman@mac.com>, Gabor Grothendieck <ggrothendieck@gmail.com>, Søren Højsgaard <Soren.Hojsgaard@agrsci.dk>, Ayal Pinkus <apinkus@xs4all.nl>

Maintainer G. Grothendieck <ggrothendieck@gmail.com>

Encoding latin1

Description An interface to the yacas computer algebra system.

Depends R (>= 2.5.1), XML (>= 2.1-0)

SystemRequirements yacas (= 1.0.63) # instructions on home page

License GPL-2

URL <http://ryacas.googlecode.com>

Repository CRAN

Date/Publication 2009-03-03 08:11:12

R topics documented:

Ryacas-package	2
bodyAsExpression	3
Eval	4
runYacas	4
Sym	5
yacas	6
yacasInstall	9
yacasTranslations	10
yacmode	11
Index	13

Description

Ryacas allows one to use the yacas computer algebra package entirely from within R. It takes an R expression, an R one line function or a yacas string and returns an R expression or a variety of other formats. It can be used for symbolic mathematics, exact arithmetic, ASCII pretty printing and R to TeX conversions. The main command is `yacas` and `?yacas` provides some information on installation and startup.

Details

The following are sources of information on "Ryacas":

DESCRIPTION file	<code>library(help = Ryacas)</code>
List of demo files	<code>demo(package = "Ryacas")</code>
Demo file	<code>demo("Ryacas")</code>
Demo	<code>demo("Ryacas-PrettyPrinter")</code>
Demo	<code>demo("Ryacas-Function")</code>
Demo	<code>demo("Ryacas-Sym")</code>
Demo	<code>demo("Ryacas-Expr")</code>
List Vignettes	<code>vignette(package = "Ryacas")</code>
Vignette	<code>vignette("Ryacas")</code>
This File	<code>package?Ryacas</code>
Help files	<code>?yacas</code> , <code>?yacasTranslations</code> , <code>?yacmode</code> , <code>?Sym</code>
Help files - Windows	<code>?yacasInstall</code>
News	<code>RShowDoc("NEWS", package = "Ryacas")</code>
Acknowledgements	<code>RShowDoc("THANKS", package = "Ryacas")</code>
Wish List	<code>RShowDoc("WISHLIST", package = "Ryacas")</code>
Home page	http://code.google.com/p/ryacas/

Note

There is a note in the help file of the `yacas` command that discusses a number of installation and startup issues.

Examples

```
## Not run:
print(yacas(expression(integrate(1/x, x))))
print(yacas("Integrate(x)1/x"))
x <- Sym("x"); Integrate(1/x, x)
acos(Sym("1/2"))
## End(Not run)
```

bodyAsExpression *Get body of function as an expression.*

Description

Get body of function as an expression.

Usage

```
bodyAsExpression(x)
```

Arguments

x An R function.

Details

This function is similar to the R `body` function except that function returns a call object whereas this one returns an expression usable in Ryacas calculations.

Value

An expression.

See Also

[body](#)

Examples

```
## Not run:

# construct an R function for the Burr probability density
# function (PDF) given the Burr cumulative distribution function (CDF)
BurrCDF <- function(x, c = 1, k = 1) 1-(1+x^c)^-k

# transfer CDF to yacas
yacas(BurrCDF)

# create a template for the PDF from the CDF
BurrPDF <- BurrCDF

# differentiate CDF and place resulting expression in body
body(BurrPDF) <- yacas(expression(deriv(BurrCDF(x, c, k))))[[1]]

# test out PDF
BurrPDF(1)

## End(Not run)
```

Eval	<i>Evaluate a yacas expression.</i>
------	-------------------------------------

Description

Evaluate a yacas expression.

Usage

```
## S3 method for class 'yacas':
Eval(x, env = parent.frame(), ...)
## S3 method for class 'Sym':
Eval(x, env = parent.frame(), ...)
```

Arguments

x	Object to be evaluated.
env	Environment or list in which to perform evaluation.
...	Not currently used.

Examples

```
## Not run:
Eval(yacas(expression(x*x)), list(x=2))

# same
x <- 2
Eval(yacas(expression(x*x)))
## End(Not run)
```

runYacas	<i>Run a yacas session directly.</i>
----------	--------------------------------------

Description

This command is typically used without arguments. It simply runs an ordinary interactive yacas session. Note that this command is not used to spawn the yacas server but just to interact directly with yacas independently of R.

Usage

```
runYacas(method = "system", yacas.args = "", yacas.init = "")
```

Arguments

<code>method</code>	Can be "system" or "server". Normally not used.
<code>yacas.args</code>	Can be used to specify the yacas command line arguments. Normally this is not used.
<code>yacas.init</code>	Can be used to specify the yacas command line <code>--init</code> argument. Normally this is not used.

Value

No return value.

Examples

```
## Not run:
yacasRun()
## End(Not run)
```

Sym

Sym

Description

The Symbol interface to yacas.

Usage

```
Sym(...)  
Expr(x)
```

Arguments

<code>x</code>	An R expression.
<code>...</code>	An R character string or object that can be coerced to a character string.

Details

An object of class "Sym" is internally a yacas character string. An object of class "Expr" is internally an R expression. One can combine such objects using the Math and Ops R operators (see `help(Math)` and `help(Ops)` for a list). Also there are methods for a number of R generics: `as.character.Sym`, `as.expression.Sym`, `determinant.Sym`, `deriv.Sym` and `print.Sym` and yacas-oriented functions: `Clear`, `Conjugate`, `Expand`, `Factor`, `Factorial`, `I`, `Identity`, `Infinity`, `Integrate`, `Inverse`, `InverseTaylor`, `Limit`, `List`, `N`, `Newton`, `Pi`, `Precision`, `PrettyForm`, `PrettyPrinter`, `Set`, `Simplify`, `Solve`, `Subst`, `Taylor`, `TeXForm`, `Ver` and " all of which have the same meaning as the corresponding yacas commands. Try `vignette("Ryacas-Sym")` for many examples.

Value

Sym returns a "Sym" object and Expr returns an "Expr" object.

Note

Currently the only Expr methods implemented are as.character.Expr, deriv.Expr, Math.Expr, Ops.Expr and print.Expr.

Examples

```
## Not run:
x <- Sym("x")
x*x
Integrate(x*x, x)
Sym("

acos(Sym("1/2"))

y <- Exprq(x)
y*y
deriv(y*y, y)
Exprq(acos(1/2))
## End(Not run)
```

yacas

yacas interface

Description

Interface to the yacas computer algebra system.

Usage

```
## S3 method for class 'character':
yacas(x, verbose = FALSE, method,
retclass = c("expression", "character", "unquote"), addSemi = TRUE, ...)
## S3 method for class 'expression':
yacas(x, ...)
## S3 method for class 'function':
yacas(x, ...)
## S3 method for class 'formula':
yacas(x, ...)
## S3 method for class 'yacas':
yacas(x, ...)
```

Arguments

<code>x</code>	A yacac character string or an R expression without terminating semicolon to be processed by yacac.
<code>verbose</code>	A logical value indicating verbosity of output or "input" to only show input to yacac but not output from yacac or "output" to only show output from yacac but not input to yacac.
<code>method</code>	method used to communicate with yacac. If "socket" is specified then the same yacac session is used on a sequence of calls. If "system" is specified then a new instance of yacac is used just for the period of that call. "system" does not require that the system be configured to support telnet/sockets and so may be useful in some instances. If no value is specified the default is taken from <code>getOption("yacac.method")</code> and if that is not specified "socket" is used. "socket" and "system" may be abbreviated.
<code>addSemi</code>	If TRUE a semicolon is added to the character string sent to yacac. This can be set to FALSE if its known that the character string already has a trailing semicolon. It is ignored if <code>retclass="expression"</code> .
<code>retclass</code>	The class of the first component of the yacac structure. It defaults to "expression" but may be specified as "character" or "unquote". "unquote" is the same as "character" except that if the character string returned would have otherwise had quotes in the first and last positions then they are stripped.
<code>...</code>	Additional arguments ultimately passed down to <code>yacac.character</code> .

Details

The user supplies an R expression, an R function name corresponding to a function with a single line body, a formula or a yacac input string. In the case of a formula it is regarded as an expression represented by the right hand side of the formula while the left hand side, if any, is ignored.

Value

An R object of class "yacac" is returned. If `PrettyPrinter("OMForm")` is in effect, which it is by default, then the first component is an R expression and the `OMForm` component contains OpenMath XML code. In other cases the first component is NULL and the `YacacForm` or `PrettyForm` components have display information.

Note

Windows Installation. On Windows one can install `Ryacac` by issuing the commands:

```
install.packages("Ryacac", dep = TRUE)
library(Ryacac)
yacacInstall()
```

or by using the `Packages | Install package(s)` menu in place of the first command. The second command downloads `scripts.dat` and `yacac.exe` from the internet and installs them into `R_HOME/library/Ryacac/yacdir` where `R_HOME` is the location of your R installation. Normally the default locations of yacac, its initialization file and the scripts file are sufficient but,

if necessary, they can be overridden via the environment variables: `YACAS_HOME`, `YACAS_INIT` and `YACAS_SCRIPTS`. The `YACAS_INVOKE_STRING` environment variable discussed in the next section overrides all three of these.

All OS Installation. The `YACAS_INVOKE_STRING` environment variable can be used to override the invocation string for yacas. Normally it is not used. If it does need to be used then a typical use might be:

```
library(Ryacac)
# only need to do the file.copy command once
file.copy(system.file("yacdir/R.yc", package = "Ryacac"), "~/.yacsrc")
# this needs to be done once per session
Sys.setenv(YACAS_INVOKE_STRING = "yacac -pc -server 9734")
demo(Ryacac) # test it out
```

`yacmode`. There is also a utility `yacmode` which is called without arguments and just turns R into a terminal into yacas until one quits out of it (and back to R) by entering `stop`, `end`, `quit`, `exit` or `e`.

Startup. `yacas` starts up when `yacasStart()` is called or the first time `yacas` is called. `yacas` is shut down when `yacasStop()` is called or when the package is detached using the `detach()` R command. On Windows, when `yacas` is shut down, the `yacas` process is terminated on Windows XP Pro but not on other versions of Windows. In those cases there will be a dangling process that the user must terminate manually.

Translation. The translation process occurs in several steps. If the input to the `yacas` function is an expression then it is translated to a valid yacas character string (otherwise, it is sent to `yacas` unprocessed). Yacas then processes the string and if `retclass="expression"` it is translated back to an R expression (otherwise it is sent back unprocessed). Examples of translations are:

R	yacas
$\sin(x)$	<code>Sin(x)</code>
$\text{deriv}(\sin, x)$	<code>Deriv(x)Sin(x)</code>
$\log(x)$	<code>Ln(x)</code>

References

<http://yacac.sourceforge.net>

Examples

```
## Not run:
yacac(expression(Factor(x^2-1)))
exp1 <- expression(x^2 + 2 * x^2)
exp2 <- expression(2 * exp0)
exp3 <- expression(6 * pi * x)
exp4 <- expression((exp1 * (1 - sin(exp3))) / exp2)
print(yacac(exp4))

print(yacac("Version()")) # yacas version

# see demo("Ryacac-Function")
```

```
## End(Not run)
```

```
yacasInstall      Install yacas files needed by Ryacas
```

Description

Download the `script.dat` and `yacas.exe` files needed by Ryacas.

Usage

```
yacasInstall(url = "http://ryacas.googlecode.com/files/yacas-1.0.63.zip",
             overwrite = FALSE)
yacasFile(filename = c("yacas.exe", "scripts.dat", "R.y"),
          slash = c("\\", "/"))
```

Arguments

<code>url</code>	URL of a zip file containing <code>yacas.exe</code> and <code>scripts.dat</code> .
<code>overwrite</code>	If TRUE then existing files are overwritten.
<code>filename</code>	Name of file whose full path is wanted.
<code>slash</code>	slash style to use on output pathname.
<code>...</code>	Additional arguments passed to <code>download.file</code> . In most cases this is not used.

Details

`yacasInstall` downloads a zip file and extracts `yacas.exe` and `scripts.dat` installing them into `yacasFile("yacas.exe")` and `yacasFile("scripts.dat")`.

Value

`yacasInstall` has no return value.

`yacasFile` returns the full pathname of the indicated yacas file or the location where it would be if it were installed. For `yacas.exe` this is the contents of the environment variable `YACAS_HOME` or if that is not defined then it is the folder `system.file(package = "Ryacas", "yacadir")`. For `scripts.dat` this is the contents of the environment variable `YACAS_SCRIPTS`, or if that is not defined, it is the folder where `yacas.exe` is located or would be located. For `R.y` this is the contents of the environment variable `YACAS_INIT`, or if that is not defined, it is located in `system.file(package = "Ryacas", "yacadir")`. Note that `R.y` is included with Ryacas since it is a text file but `scripts.dat` is not included since it is a binary file which is why their default locations differ.

`yacasCheck` returns 0 if the yacas files, `yacas.exe` and `scripts.dat` were found in `yacasFile("yacas.exe")` and `yacasFile("scripts.dat")`, -1 if they were not found and 1 if they were found but have the wrong file size. If the user specifies the `YACAS_INVOKE_STRING` environment variable then it will not be able to perform the check in which case NA is returned.

Note

These functions are for Windows systems only. For other platforms these environment variables are not available and the user must install yacas manually prior to installing Ryacas.

Examples

```
## Not run:
Sys.getenv("YACAS_INVOKE_STRING")
Sys.getenv("YACAS_HOME")
Sys.getenv("YACAS_SCRIPTS")
system.file(package = "Ryacas", "yacdir")
yacasFile("yacac.exe")
yacacFile("scripts.dat")
yacacInstall()
## End(Not run)
```

yacasTranslations *Yacas translations*

Description

Translations from R to the yacas computer algebra system.

Note

The translation process occurs in several steps. If the input to the `yacas` function is an expression then it is translated to a valid yacas character string (otherwise, it is sent to yacas unprocessed). Yacas then processes the string and if `retclass="expression"` it is translated back to an R expression (otherwise it is sent back unprocessed). Currently supported translations are:

CONSTANTS

R	yacas
=	=====
pi	Pi

OPERATORS

R	yacas
=	=====
7 %% 3	Mod(7, 3)
7 %/% 3	Div(7, 3)

FUNCTIONS

R	yacas
---	-------

=	=====
sin(x)	Sin(x)
cos(x)	Cos(x)
tan(x)	Tan(x)
asin(x)	ArcSin(x)
acos(x)	ArcCos(x)
atan(x)	ArcTan(x)
exp(x)	Exp(x)
sqrt(x)	Sqrt(x)
log(x)	Ln(x)
choose(n, k)	Bin(n, k)
gamma(x)	Gamma(x)
deriv(sin, x)	Deriv(x)Sin(x)
integrate(f, a, b)	Integrate(x, a, b)f(x)
list()	List()
factorial(n)	n!

Note the Limit example in demo(Ryacac) for adding translations on the fly.
The complete table under development.

Author(s)

Rob J Goedman

References

<http://yacac.sourceforge.net>

yacmode

yacmode interface

Description

Interactive interface to the yacac

Usage

yacmode ()

Details

The user types valid yacac input and presses return. Type 'quit' to return to R prompt.

Value

Output of yacac is returned.

Note

Note that command recall will recall previous R commands, not previous yacas input. Yacas is given a limited amount of time to complete, otherwise '[1] CommandLine(1) : User interrupted calculation' is returned. E.g. `Taylor(x,0,5) 1/(1+x)` will work, but `Taylor(x,0,12) 1/(1+x)` is likely to take too long.

References

<http://yacas.sourceforge.net>

Examples

```
## Not run:  
yacmode()  
  (x+y)^3-(x-y)^3  
  Simplify(  
  1  
  ## End(Not run)
```

Index

- *Topic **programming**
 - Ryacas-package, 2
- *Topic **symbolmath**
 - bodyAsExpression, 3
 - Eval, 4
 - runYacas, 4
 - Sym, 5
 - yacas, 6
 - yacasInstall, 9
 - yacasTranslations, 10
 - yacmode, 11
- %Where% (Sym), 5

- addSemi (yacas), 6
- as.character.Expr (Sym), 5
- as.character.Sym (Sym), 5
- as.character.yacas (yacas), 6
- as.Expr.formula (Sym), 5
- as.expression.Sym (Sym), 5
- as.expression.yacas (yacas), 6
- as.language (bodyAsExpression), 3
- as.Sym (Sym), 5

- body, 3
- bodyAsExpression, 3

- Clear (Sym), 5
- Conjugate (Sym), 5

- deriv.Expr (Sym), 5
- deriv.Sym (Sym), 5
- determinant.Expr (Sym), 5
- determinant.Sym (Sym), 5

- Eval, 4
- Expand (Sym), 5
- Expr (Sym), 5
- Exprq (Sym), 5

- Factor (Sym), 5
- Factorial (Sym), 5

- haveYacas (yacas), 6

- I (Sym), 5
- Identity (Sym), 5
- Infinity (Sym), 5
- Integrate (Sym), 5
- Inverse (Sym), 5
- InverseTaylor (Sym), 5
- isConnection (yacas), 6

- Limit (Sym), 5
- List (Sym), 5

- Math.Expr (Sym), 5
- Math.Sym (Sym), 5

- N (Sym), 5
- Newton (Sym), 5

- OpenMath2R (Sym), 5
- Ops.Expr (Sym), 5
- Ops.Sym (Sym), 5
- Ops.yacas.symbol (Sym), 5

- Pi (Sym), 5
- Precision (Sym), 5
- PrettyForm (Sym), 5
- PrettyPrinter (Sym), 5
- print.Expr (Sym), 5
- print.Sym (Sym), 5
- print.yacas (Sym), 5

- root (yacasTranslations), 10
- runYacas, 4
- Ryacas-package, 2

- Set (Sym), 5
- Simplify (Sym), 5
- Solve (Sym), 5
- Subst (Sym), 5
- Sym, 5

SymExpr (*Sym*), 5

Taylor (*Sym*), 5

TeXForm (*Sym*), 5

trans (*Sym*), 5

transtab (*Sym*), 5

Ver (*Sym*), 5

yacas, 2, 6

yacas.symbol.value (*Sym*), 5

yacasFile (*yacasInstall*), 9

yacasInstall, 9

yacasInvokeString (*yacas*), 6

yacasStart (*yacas*), 6

yacasStop (*yacas*), 6

yacasTranslations, 10

yacmode, 11

yAssignFunction (*yacas*), 6

yDeriv (*Sym*), 5

yFactorial (*Sym*), 5

yIntegrate (*Sym*), 5

yLimit (*Sym*), 5

ynext (*yacas*), 6

yparse (*yacas*), 6

yrewrite (*Sym*), 5

ySequence (*yacas*), 6

ysub (*yacas*), 6

yUnlist (*Sym*), 5