

Package ‘Ryacas0’

July 21, 2025

Version 0.4.4

Title Legacy 'Ryacas' (Interface to 'Yacas' Computer Algebra System)

Maintainer Mikkel Meyer Andersen <mik1@math.aau.dk>

Encoding UTF-8

Description A legacy version of 'Ryacas', an interface to the 'yacas' computer algebra system (<<http://www.yacas.org/>>).

Depends R (>= 3.3.0)

Imports methods, Rcpp (>= 0.12.0), stats, settings, xml2

LinkingTo Rcpp

Suggests devtools, exams, knitr, Matrix, pkgload, rmarkdown, igraph, testthat

License GPL-2

URL <https://github.com/r-cas/ryacas0>, <http://www.yacas.org>

BugReports <https://github.com/r-cas/ryacas0/issues>

RoxygenNote 7.2.3

VignetteBuilder knitr

NeedsCompilation yes

Author Mikkel Meyer Andersen [aut, cre, cph],
Rob Goedman [aut, cph],
Gabor Grothendieck [aut, cph],
Søren Højsgaard [aut, cph],
Grzegorz Mazur [aut, cph],
Ayal Pinkus [aut, cph],
Nemanja Trifunovic [cph] (UTF-8 part of yacas
(src/yacas/include/yacas/utf8*))

Repository CRAN

Date/Publication 2023-01-12 09:50:05 UTC

Contents

Ryacas0-package	2
as.Sym.character	3
as.Sym.matrix	4
bodyAsExpression	4
Eval	5
getSyms	6
get_output_width	6
root	7
Ryacas_options	7
set_output_width	8
stripvar	8
syacas	9
Sym	9
yacas	10
yacasTranslations	13
yacas_evaluate	14
yacmode	15
y_ls	16
Index	17

Ryacas0-package	<i>R interface to yacas computer algebra package</i>
-----------------	--

Description

Ryacas0 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 "Ryacas0":

DESCRIPTION file	library(help = Ryacas0)
List of demo files	demo(package = "Ryacas0")
Demo file	demo("Ryacas0")
Demo	demo("Ryacas0-PrettyPrinter")
Demo	demo("Ryacas0-Function")
Demo	demo("Ryacas0-Sym")
Demo	demo("Ryacas0-Expr")
List Vignettes	vignette(package = "Ryacas0")
Vignette	vignette("Ryacas0")
This File	package?Ryacas0

Help files	?yacas, ?yacasTranslations, ?yacmode, ?Sym
Help files - Windows	?yacasInstall
News	RShowDoc("NEWS", package = "Ryacas0")
Acknowledgements	RShowDoc("THANKS", package = "Ryacas0")
Wish List	RShowDoc("WISHLIST", package = "Ryacas0")
Home page	https://github.com/r-cas/ryacas0/

Note

There is a note in the help file of the [yac](#) command that discusses a number of installation and startup issues.

Examples

```
print(yacas(expression(integrate(1/x, x))))
print(yacas("Integrate(x)1/x"))
x <- Sym("x"); Integrate(1/x, x)
acos(Sym("1/2"))
```

as.Sym.character	<i>Convert character vector to yacas object</i>
------------------	---

Description

Simple and raw conversion to yacas

Usage

```
## S3 method for class 'character'
as.Sym(x, ...)
```

Arguments

x	An R character vector.
...	Not used

Examples

```
x <- c("a", "2", "4", "c", "d", "6")
x
y <- as.Sym(x)
y
Eval(y, list(a = 3, c = 3, d = 3))
```

as.Sym.matrix	<i>Convert character matrix to yacas object</i>
---------------	---

Description

Simple and raw conversion to yacas

Usage

```
## S3 method for class 'matrix'
as.Sym(x, ...)
```

Arguments

x	An R character matrix.
...	Not used

Examples

```
x <- matrix(c("a", "2", "4", "c", "d", "6"), 3, 2)
x
y <- as.Sym(x)
y
Eval(y, list(a = 3, c = 3, d = 3))
```

bodyAsExpression	<i>Get body of function as an expression.</i>
------------------	---

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

```
# 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)
```

 Eval

Evaluate a yacas expression.

Description

Evaluate a yacas expression.

Usage

```
Eval(x, env = parent.frame(), ...)
```

Arguments

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

Examples

```
Eval(yacas(expression(x*x)), list(x=2))

# same
x <- 2
Eval(yacas(expression(x*x)))
```

getSyms	List Sym() objects
---------	--------------------

Description

Lists all Sym() objects in the global environment (.GlobalEnv)

Usage

```
getSyms(all.names = FALSE)
```

Arguments

all.names a logical value. If TRUE, all object names are returned. If FALSE, names which begin with a . are omitted.

Examples

```
getSyms()
xs <- Sym("x")
getSyms()
```

get_output_width	Get width of yacas output
------------------	---------------------------

Description

Get width of yacas output

Usage

```
get_output_width()
```

root	<i>Root function</i>
------	----------------------

Description

The y 'th root of x , i.e. $x^{(1/y)}$.

Usage

```
root(x, y)
```

Arguments

x	Number to take y 'th root of
y	Root

Value

The root y 'th root of x

Ryacas_options	<i>Set or get options for the Ryacas package</i>
----------------	--

Description

Set or get options for the Ryacas package

Usage

```
Ryacas_options(...)
```

Arguments

...	Option names to retrieve option values or [key] = [value] pairs to set options.
-----	---

Supported options

The following options are supported

- `module_matvec_enabled` (default TRUE): Print yacas `List()`'s as vectors and `List(List(), ...)`'s as matrices.
- `prettyform_default` (default FALSE): Print yacas as `PrettyForm()` as default.

set_output_width	<i>Set width of yacas output</i>
------------------	----------------------------------

Description

Set width of yacas output

Usage

set_output_width(w)

Arguments

w	Width in number of characters
---	-------------------------------

stripvar	<i>Removes part of expression containing variable</i>
----------	---

Description

Yacas' Solve(eq, x) can return e.g. $x == \text{expr}$ and $\{x == \text{expr1}, x == \text{expr2}, \dots\}$. Some usages are easier if the initial $x ==$ part is removed. This is the purpose of this function.

Usage

stripvar(expr, var)

Arguments

expr	Expression where $x == \text{expr}$ should be replaced to expr
var	Name of variable, e.g. x

Value

Expression with left-hand side removed

syacas	<i>yacas interface – silent version</i>
--------	---

Description

Similar to `yacas()` but silent. This can be useful when working with yacas directly.

Usage

`syacas(x, ...)`

Arguments

- | | |
|------------------|---|
| <code>x</code> | A yacas character string or an R expression without terminating semicolon to be processed by yacas. |
| <code>...</code> | Additional arguments ultimately passed down to <code>yacas.character</code> . |

See Also

`yacas`

Sym	<i>Sym</i>
-----	------------

Description

The Symbol interface to yacas.

Usage

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

Arguments

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

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`, `Transpose`, `Ver` and "%Where%" all of which have the same meaning as the corresponding yacas commands. Try `vignette("Ryac-Sym")` for many examples.

Get Sym objects with `getSyms()`.

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`.

See Also

[as.Sym.matrix\(\)](#)

Examples

```
x <- Sym("x")
x*x
Integrate(x*x, x)
Sym("%") %Where% list(x = 10)

acos(Sym("1/2"))

y <- Exprq(x)
y*y
deriv(y*y, y)
Exprq(acos(1/2))
```

yacas

yacas interface

Description

Interface to the yacas computer algebra system.

Usage

```

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

```

Arguments

<code>x</code>	A yacas character string or an R expression without terminating semicolon to be processed by yacas.
<code>...</code>	Additional arguments ultimately passed down to <code>yacas.character</code> .
<code>verbose</code>	A logical value indicating verbosity of output or "input" to only show input to yacas but not output from yacas or "output" to only show output from yacas but not input to yacas.
<code>method</code>	method used to communicate with yacas. If "socket" is specified then the same yacas session is used on a sequence of calls. If "system" is specified then a new instance of yacas 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("yacas.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 yacas. 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 yacas 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 and last positions then they are stripped.

Details

The user supplies an R expression, an R function name corresponding to a function with a single line body, a formula or a yacas 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.

Note the silent version `syacas()`.

Value

An R object of class "yacas" 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 Open-Math XML code. In other cases the first component is NULL and the YacasForm or PrettyForm components have display information.

Generally an expression. Refer to details.

Note

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

```
install.packages("Ryacas", dep = TRUE)
library(Ryacas)
yacasInstall()
```

or by using the Packages | Install package(s) menu in place of the first command. The second command downloads `scripts.dat` and `yacas.exe` from the internet and installs them into `R_HOME/library/Ryacas/yacdir` where `R_HOME` is the location of your R installation.

Normally the default locations of yacas, 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(Ryacas)
# only need to do the file.copy command once
file.copy(system.file("yacdir/R.y", package = "Ryacas"), "~/yacsrc")
# this needs to be done once per session
Sys.setenv(YACAS_INVOKE_STRING = "yacas -pc -server 9734")
demo(Ryacas) # 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
<code>sin(x)</code>	<code>Sin(x)</code>
<code>deriv(sin, x)</code>	<code>Deriv(x)Sin(x)</code>
<code>log(x)</code>	<code>Ln(x)</code>

References

<http://www.yacas.org/>

Examples

```

yacas(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(yacas(exp4))

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

# see demo("Ryacac-Function")

```

yacasTranslations	<i>Yacas translations</i>
-------------------	---------------------------

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)
$7 \% \wedge \% 3$	Div(7, 3)

FUNCTIONS

R	yacas
=	=====
sin(x)	Sin(x)
cos(x)	Cos(x)

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

Note the Limit example in demo(Ryacac0) for adding translations on the fly.
The complete table under development.

Author(s)

Rob J Goedman

References

<http://www.yacas.org/>

<code>yacas_evaluate</code>	<i>Evaluate yacas expression</i>
-----------------------------	----------------------------------

Description

This is a low-level function for evaluating yacas expression represented as string.

Usage

```
yacas_evaluate(expr)
```

Arguments

<code>expr</code>	Yacas expression
-------------------	------------------

Value

Result of evaluating `expr` by yacas in OpenMath format and side-effects of the evaluation

Examples

```
yacas_evaluate("D(x)Sin(x^2)")
```

yacmode*yacmode interface*

Description

Interactive interface to the yacas

Usage

```
yacmode(enable_history = TRUE)
```

Arguments

`enable_history` Use R history such that previous yacas commands can be used. Default is TRUE.

Details

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

Value

Output of yacas is returned.

invisible NULL

Note

Note that command will use R `history()` and modify it by default. 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://www.yacas.org/>

Examples

```
## Not run:
yacmode()
(x+y)^3-(x-y)^3
Simplify(%)
q

## End(Not run)
```

`y_ls`*Get Yacas variables*

Description

Get Yacas variables

Usage

`y_ls()`

Value

Vector of variables defined in yacas

Index

- * **programming**
 - Ryacas0-package, 2
- * **symbolmath**
 - bodyAsExpression, 4
 - Eval, 5
 - stripvar, 8
 - Sym, 9
 - yacas, 10
 - yacasTranslations, 13
 - yacmode, 15
- %Where% (Sym), 9
- addSemi (yacas), 10
- as.character.Expr (Sym), 9
- as.character.Sym (Sym), 9
- as.character.yacas (yacas), 10
- as.Expr.formula (Sym), 9
- as.expression.Sym (Sym), 9
- as.expression.yacas (yacas), 10
- as.language (bodyAsExpression), 4
- as.Sym (Sym), 9
- as.Sym.character, 3
- as.Sym.matrix, 4
- as.Sym.matrix(), 10
- body, 5
- bodyAsExpression, 4
- CharacteristicEquation (Sym), 9
- Clear (Sym), 9
- Conjugate (Sym), 9
- deriv.Expr (Sym), 9
- deriv.Sym (Sym), 9
- determinant.Expr (Sym), 9
- determinant.Sym (Sym), 9
- EigenValues (Sym), 9
- Eval, 5
- Expand (Sym), 9
- Expr (Sym), 9
- Exprq (Sym), 9
- Factor (Sym), 9
- Factorial (Sym), 9
- FindRoots (Sym), 9
- get_output_width, 6
- getSyms, 6
- I (Sym), 9
- Identity (Sym), 9
- Infinity (Sym), 9
- Integrate (Sym), 9
- Inverse (Sym), 9
- InverseTaylor (Sym), 9
- Limit (Sym), 9
- List (Sym), 9
- Math.Expr (Sym), 9
- Math.Sym (Sym), 9
- N (Sym), 9
- Newton (Sym), 9
- OpenMath2R (Sym), 9
- Ops.Expr (Sym), 9
- Ops.Sym (Sym), 9
- Ops.yacas.symbol (Sym), 9
- Pi (Sym), 9
- Precision (Sym), 9
- PrettyForm (Sym), 9
- PrettyPrinter (Sym), 9
- print.Expr (Sym), 9
- print.Sym (Sym), 9
- print.yacas (Sym), 9
- root, 7
- Ryacas0-package, 2
- Ryacas_options, 7

Set (Sym), 9
set_output_width, 8
Simplify (Sym), 9
Solve (Sym), 9
stripvar, 8
Subst (Sym), 9
syacas, 9
syacas(), 11
Sym, 9
SymExpr (Sym), 9

Taylor (Sym), 9
TeXForm (Sym), 9
trans (Sym), 9
Transpose (Sym), 9
transtab (Sym), 9

Ver (Sym), 9

y_ls, 16
yacas, 3, 10
yacas(), 9
yacas.symbol.value (Sym), 9
yacac_evaluate, 14
yacacTranslations, 13
yacmode, 15
yAssignFunction (yacac), 10
yDeriv (Sym), 9
yFactorial (Sym), 9
yIntegrate (Sym), 9
yLimit (Sym), 9
ynext (yacac), 10
yparse (yacac), 10
yrewrite (Sym), 9
ySequence (yacac), 10
ysub (yacac), 10
yUnlist (Sym), 9