NAME

ries – find algebraic equations, given their solution

SYNOPSIS

ries [**-l***n*] [**-i[e]**] [**-s**] [**-x**] [**-F***n*] [**-S***sss*] [**-N***sss*] [**-O***sss*] [**-D***xxx*] [**-p***filename*] [**--extended-options** [...]] *value*

ries --find-expression [*expression* [...]]

ries --eval-expression [*expression* [...]]

DESCRIPTION

Given a number, **ries** searches for algebraic equations in one variable that have a solution (root) near that number. It avoids trivial or reducible solutions like ''*x*/*x* = 1''. If *value* is an integer, **ries** can find an exact solution expressed in terms of single-digit integers.

For example, if you supply the value 2.5063, the first part of **ries**'s output will resemble the following:

```
$ ries 2.5063

    Your target value: T = 2.5063              mrob.com/ries

            2 x = 5              for x = T - 0.0063       {49}
            8 x = e^3            for x = T + 0.00439212   {66}
            x^2 = 2 pi           for x = T + 0.000328275  {55}
            x^x = 1+9            for x = T - 0.000115854  {69}
          x^2+e = 9              for x = T + 3.56063e-05  {63}
        ln(6) x = sqrt(pi)+e     for x = T + 2.73037e-05  {93}
          x/4+1 = 4,/7           for x = T + 6.24679e-06  {91}
 sinpi(ln(x))^2 = 1/(5 pi)       for x = T + 2.75665e-06  {92}
```

The output gives progressively ''more complex'' solutions (as described below) that come progressively closer to matching your number. There are four columns: equations in symbolic form (two columns of expressions with '=' in the middle), solution of equation (value of *x* expressed as *T* plus a small error term), and total complexity score (described below).

Each match is checked by solving for *x* using the Newton-Raphson method, and the closeness of the match is judged by the difference between the root (the value of *x* for which the two sides are equal) and your target value *T*.

Options allow complete control over what symbols, constants and functions are used in solutions, or to limit solutions to integer, rational, constuctible, or algebraic values.

OPTIONS

Options must be separate: '`ries -l1 -i -Ox 27`', not '`ries -l1iOx 27`'.

-p*name* Profile (or Parameters): Load one or more options from file *name*. **-p***name* is equivalent to **--include** *name*, which is described in the EXTENDED OPTIONS section below. **-p** alone (with no *name*) has special meanings, also described under **--include**.

-l*n* Level: Specifies the level of the search (default 2). With each increment of **-l**, **ries** will search about 10 times as many equations, use 3.5 times as much memory and take at least 4 times as long.

Use higher levels to add more factors of 10. The level can be fractional or negative. Here are typical figures, measured on a Core i7 at 3.2 GHz (using only one thread) invoked by the command **ries -l*n* 2.5063141592653589** for different values of searchlevel *n*:

	memory	equations tested	digits	run time
-l0	1.2M	89,400,000	6+	0.025 sec
-l1	4.0M	932,000,000	7+	0.08 sec
-l2	14 M	11,400,000,000	8+	0.33 sec
-l3	45 M	134,000,000,000	9+	1.8 sec
-l4	158M	1,600,000,000,000	11+	8.8 sec
-l5	490M	15,000,000,000,000	12+	37.1 sec
-l6	1.7G	184,000,000,000,000	13+	190 sec

(These times are a little quicker than a 2.33-GHz Core 2 Duo; on a 733-MHz Pentium 3, the times were about 5 times longer. If compiled for an environment with 32-bit pointers, memory usage figures are about 20% lower. **ries** also works on much older and smaller systems, and can test billions of equations in less than a minute on 1990's hardware.)

Use a fractional argument (like **-l5.5**) for more precise control of how much memory **ries** will use before stopping its search. When free memory is exhausted; performance will degrade significantly and **ries** might exit, depending on your operating system. Under Linux and Mac OS, **ries** keeps running but the system slows to a crawl. If you don't know what your OS will do, be careful before running **ries** with higher levels. In extreme cases your computer's response might slow down so much that you are unable to save your work in other applications.

The memory limits are not reached nearly as quickly when the symbolset is greatly limited with **-S**, **-O** and **-N** or when **-i** is specified. **-i** in particular should allow about two more levels in any given amount of memory. Large arguments tend to lengthen runtime: for example, `ries -l4 1058073667` takes about three times as long as `ries -l4 1.058073667`.

Options to Select Symbols

Several options are used to choose which symbols (constants, operations, and functions) **ries** is allowed to use when searching for equations.

-N*sss* Never use these: **-N** followed by one or more characters specifies symbols (constants and operators) that **ries** should not use in its equations. The symbols are as follows:

- 1-9 The integers 1 through 9. (**ries** constructs all larger integers from combinations of these.)
- p pi = 3.14159...
- e e = 2.71828...
- f phi = (1+sqrt(5))/2 = 1.61803...
- n Negative
- r Reciprocal
- s Squared
- q Square root
- S C Sine, Cosine
- T Tangent

A Two-argument Arctangent: arctan(y,x) is the angle of the ray from the origin through the point (x,y)

l ln (natural logarithm, also called log)

E e to the power of x

+ - Add, Subtract

* / Multiply, Divide

ˆ Power: 2 ˆ 3 = 8

v Root (the "v" resembles part of the radical symbol): 3 v 27 = cube root of 27

L Logarithm to base A of B

W Lambert W function. Only available if using the stand-alone maths library, described in the section "STAND-ALONE MATHS LIBRARY" below. In addition, one must explicitly choose it with **–EW**.

There are lots of potential uses for **–N**. For example, if you invoke **ries** on a small irrational number, you might get several solutions that involve the unary and binary logarithm operators 'ln' (natural logarithm) and 'log_' (log to base A of B). If you decide you aren't interested in such solutions you can just add **-NlL** to your command line, and all such solutions will be skipped.

If you are checking an unknown number that you found in the context of some larger problem, you probably have some idea what constants and operators may be involved, or not involved, in the phenomenon that produced your number. Use **-N** to rule out functions you don't think are relevant.

Note that **ries** will often run considerably slower when you limit it to a very small set of symbols, mainly because it cannot use its optimization rules (described below under ALGORITHM). Also, with fewer symbols the average length of expressions is longer, and that makes the search slower.

-S*sss* permitted Symbol Set: Specifies a symbol set, as with **–N**, but has the opposite effect: *only* these symbols will be used. A **-S** also cancels any **–E**, **–N** or **–O** options that were given; meaning that if you wish to combine these options, the **–S** should come first.

–S can be used to solve those old problems of the sort "How can the number 27 be expressed using only the four basic operators, powers and square roots, and the digit 4?" The answer is given by:

```
ries '-S+-*/^q4' 27 --one-sided
```

(The **--one-sided** option is described below). To solve the same problem using the **-N** option, you'd need something like:

```
ries -Npef12356789rslLvE 27 --one-sided -l4
```

If you give the **–S** option with no symbols, **ries** will display a table of all available symbols (as modified by any **-E**, **-N**, **-O** and **-S** options that precede it) with their definition and weights. This lone **–S** can be given along with a normal **–S** option, but in any case **ries** will exit after showing the table.

-E*sss* Enable: Enables (or re-enables) the use of symbols that may have been disabled by an earlier **-S** or **-N** option. This is mainly intended for use in combination with the **--include** option. If one include file disables some symbols, this option can be used to re-enable some or all of them. It is also required for use of the Lambert W function, which is disabled by default. See "STAND-ALONE MATHS LIBRARY" for instructions and use the option **-EW** to request use of the Lambert W function in a **ries** command.

-O*sss* Only One: Specifies symbols which should appear no more than once on *each side* of the equation. This option can be combined with **-E** or **–N**, in which case they augment each other. If used with **-S** with the same symbol, the latter option takes effect.

One additional symbol is available with **-O**:

x The variable on the left-hand-side

Thus, you can use **-Ox** to limit **ries**'s output to equations that have only one '*x*' in them and are therefore easy to solve for *x* using only the most basic algebra techniques. This also makes **ries**'s output more like that of traditional expression-finders, which search for expressions equal to *x* rather than equations in *x*. Here's an example: `ries -i 16` gives the answer "$x^2 - x = 3^5$" with *x* very close to 16, because 16^2-16 is close to 3^5. `ries -i 16 -Ox` replaces that answer with "$5\,x^2 = 6^4$".

Options Limiting the Type of Solutions

Several options are used to choose what types of expression can be used in the equations that are presented as solutions.

-i Integer: Require that all expressions, and all subexpressions, must have integer values. This is primarily useful if you are searching for an exact solution for a large integer. Note that inexact solutions will still be given, but both sides of the equation will be integers. An example of this is "$2\,x = 7^3$" where *x*=173. **-i** is interpreted as **-r** (described below) if the supplied target value is not an integer.

-ie Integer, Exact. Like **-i**, but exits after reporting an exact match (if found). This equivalent to **-i** combined with **--min-match-distance 0**. To not report any inexact matches at all, use **--max-match-distance 0** (described in more detail below).

-r Rational: Require that all equations have a single *x* and that all subexpressions not involving *x* are rational (an exact ratio of two integers). This is primarily useful if you are searching for rational approximations. This option is essentially just shorthand for using the **-N** and **-E** options to allow only addition, subtraction, multiplication and division, excluding irrational constants and transcendental functions, etc.

-re Rational, "Exact". Like **-r**, but exits after reporting an "exact" match (if found). This equivalent to **-r** combined with **--min-match-distance 0**.

Note that computers are (famously) unable to make exact calculations with fractions as simple as 1/3. To exit on a match within some "epsilon", use **--max-match-distance** with a nonzero epsilon; to reject inexact matches use **--max-match-distance** (these options are described in detail below).

-c Constructible: Require that all equations have a single *x* and that all subexpressions are "constructible" in the sense of Euclid's *Elements*, given a unit interval. This is like the **-r** option except that squares, square roots and the golden ratio *phi* are also allowed. All results will be easily solvable for *x* and will use only addition, subtraction, multiplication, division, and square roots.

If you add the option **-Ex**, more than one *x* may appear in the solution. This gives answers that, when solved for *x*, are not constructible from the unit interval, but both sides of the (unsolved) equation *are* constructible given *x* and a unit interval. For example `ries 1.3263524026321 -c -Ex` finds the solution "$x\,x^2 = 7/3$"; *x* is the cube root of 7/3 which is not itself constructible. **ries**'s answer reflects the fact that 7/3 can be constructed from its cube root (although the opposite construction is impossible).

-a Algebraic: Generate equations whose roots are "algebraic numbers". This is similar to the **-c** option, but also allows n^{th} powers and roots. The forward trigonometric functions (sine, cosine, and tangent) are allowed but their arguments will always be rational multiples of *pi*; arctangent is not allowed because for example $\tan^{-1}(1)$ is *pi*/4. More than one *x* is allowed (unless you follow **-a** with the option **-Ox**) so the equations might not be easy to solve.

-Ox Using the **-O** option (described above) with the symbol 'x' tells **ries** to limit its search to solutions that can be expressed in "closed form" using the basic constants, elementary and transcendental functions. This concept of "closed-form number" is described by Timothy Chow in his 1998 paper *What is a closed-form number?*.

-l Liouvillian: Generate equations whose roots are "EL numbers" (exponential-logarithmic), as described by Timothy Chow in his 1998 paper *What is a closed-form number?*.

Options Affecting Output Format

-s Sorta Solve, by Shifting to right-hand side: With this option, **ries** will display equations with just a single "*x*" on the left-hand side of the equal sign. It isn't "solving" the equations, but merely performing algebraic transformations to move everything except one *x* to the right-hand side: "$x(x+1) = 7$" becomes "$x = 7/(x+1)$". You can combine this option with **-Ox** to eliminate this issue, but with that option **ries** will no longer find solutions that require more than one "*x*", like "$x^x = 2$" for 1.55961.

-x X Values: Print actual values of *x* (the roots of the equations found) rather than expressing *x* as *T* plus/minus a small number, where *T* is your target number. "**--absolute-roots**" is a synonym for **-x**.

-F*n* Format: Controls the way expressions are formatted in the main output. If *n* is omitted it is 3 ("-F" for "FORTH Format"); if **-F** is not specified at all, the format will be 2. The following formats are available; each shows the output of `ries 1.506591651 -Fn`:

Format 0: Compressed FORTH-like postfix format: Each operator and constant is just a single symbol. The symbols are as listed above under the **-N** option.

```
     x1- = 2r        for x = T - 0.00659165  {50}
     xlr = 6q        for x = T - 0.00241106  {62}
     x4^ = p2+       for x = T - 0.000766951 {68}
    x1+s = p2*       for x = T + 3.66236e-05 {69}
```

Format 1: Infix format, but with single-letter symbols. If this format is specified, a table of symbols will be printed after the main table of results. The rest of the expression syntax is the same as the normal format. For example, ‘‘q(l(*x*)) = p-1’’ means ‘‘sqrt(ln(*x*)) = pi - 1’’.

```
     x-1 = 1/2       for x = T - 0.00659165  {50}
  1/l(x) = q(6)      for x = T - 0.00241106  {62}
     x^4 = 2+p       for x = T - 0.000766951 {68}
 (x+1)^2 = 2.p       for x = T + 3.66236e-05 {69}
```

Format 2: Standard infix expression format (this is the default).

```
     x-1 = 1/2       for x = T - 0.00659165  {50}
 1/ln(x) = sqrt(6)   for x = T - 0.00241106  {62}
     x^4 = 2+pi      for x = T - 0.000766951 {68}
 (x+1)^2 = 2 pi      for x = T + 3.66236e-05 {69}
```

Format 3: Print solutions in postfix format, similar to that used in FORTH and on certain old pocket calculators. This is close to the format used internally by **ries** (to get the exact, condensed format, use *-F0*). This is intended mainly for use by scripts that use **ries** as an engine to generate equations and then perform further manipulation on them. However, this option will also help you distinguish what symbols were actually used internally to generate an answer. For example, ’squared’ and ’to the power of 2’ both show up as ’^2’ in the normal output, but in postfix they appear as "dup*" and "2 **" respectively.

```
       x 1 - = 2 recip    for x = T - 0.00659165  {50}
  x ln recip = 6 sqrt     for x = T - 0.00241106  {62}
      x 4 ** = pi 2 +     for x = T - 0.000766951 {68}
x 1 + dup* = pi 2 *       for x = T + 3.66236e-05 {69}
```

Most of the symbols used by **-F3** are self-explanatory. The nonobvious ones are: **neg** for negate, **recip** for reciprocal, **dup*** for square, **sqrt** for square root, ****** for power (A^B), **root** for Bth root of A, **logn** for logarithm (to base B) of A. For these last three, *A* is the first operand pushed on the stack and *B* is the second.

The setting of **-F** does not affect expressions displayed by the various **-D** diagnostic options (most of these use **-F0**, and **-Ds** (‘‘show your work’’) uses **-F2**).

You may use the **--symbol-names** option (described below) to redefine the appearance of formats 2 and 3.

-D*xx* Display Diagnostic/Debugging Data: A detailed understanding of the **ries** algorithms (described below) is assumed. **-D** is followed by one or more letters specifying the messages you want to see. Options **A** through **L** and **a** through **l** (except **E** and **e**) apply to the LHS and RHS respectively. For each option, the number of lines of output that you can expect from a command like `ries -l2 2.5063141592653589 -D`*x* (with *x* replaced by a single letter) is shown:

A,a **[42836; 87770]** show partial expressions that are ‘‘pruned’’ (ignored) because of arithmetic error (e.g. divide by zero)

B,b **[3173; 2714]** show partial expressions pruned for being zero, or derivative nearly zero; or outside range given by **--min-equate-value** and **--max-equate-value**

C,c **[81056; 697227]** show partial expressions pruned for being non-integer (and -i option was given); or irrational (and -r option given); etc. (sample command is `ries -l2 1047 -i -DC`)

D,d **[1751; 4350]** show partial expressions pruned because of overflow

E,e **[102356; 272746]** show expressions pruned because their value matches one already in database

F,f **[349368; 848882]** show **--canon-reduction** operations on expressions before adding to database (sample command is `ries -l2 2.50631415926 --canon-reduction nr25 -DF`)

G,g **[96112; 97337]** show expressions added to database

H,h **[409175; 816240]** show attributes of each partial expression tested

I,i **[3904331; 7759741]** show each new symbol to be added before complexity pruning

J,j **[2579116; 5102516]** show symbols skipped by complexity pruning

K,k **[257302; 558199]** show symbols skipped by redundancy and tautology rules

L,l **[61994; 114453]** show symbols skipped to obey -O option (sample command is `ries -l2 2.50631415926 '-O-+/^v*qsrlLeEpf' -DL`)

m **[10247603]** show all metastack operations

M **[46]** show memory allocation benchmarks, and enable automatic exit when memory gets slow (see **--memory-abort-threshold** option)

n **[136]** show Newton iteration values and errors if any

N **[461]** show work in detail: operator/symbol, x and dx at each step

o **[539235]** show match checks

p **[112]** show preprocessing transformations prior to conversion to infix

Q **[51]** show manipulations to remove **--canon-reduction** from equations before root-finding

q **[140]** show close matches dispatched to Newton and results of test

r **[1806085]** show results (value and derivative of operands and result) for each opcode executed

S **[100]** show solve-for-x work: displays all operations performed by the **--try-solve-for-x** option to transform an equation into "solved" form

s **[277]** show your work: displays values of each subexpression for every reported answer. Subexpressions are shown in normal (infix) syntax, which is useful in combination with **-F0** to see the postfix format used with options like **--eval-expression**

t **[11017]** show all abc-forms passed to expression generation

u **[48895]** show steps of min/max complexity ranging for each abc-form

v **[5525]** show number of expressions generated by each abc-form

w **[32922]** show details of abc-form generation (pruning, weights, etc.)

x **[91]** show all rules used (varies with the **-N**, **-O**, and **-S** options)

y **[736]** statistics and decisions made in main loop

z **[55]** initialization and other uncategorized messages

0 **[1712490]** list the entire expression database after every pass through the main loop

Of these, **-Ds** is probably the most useful and fun to look at. **-Dy** gives a nice top-level view of the statistics of the search. Most of the options that generate lots of output are useful if filtered through `grep`; this can tell you why a certain subexpression is or is not appearing in results. **-DG** and **-Dg** can be useful if you want to use **ries** to generate a massive list of expressions for processing by another program; for this reason its output uses infix notation. Most other **-D** options print subexpressions in the **-F0** terse postfix format.

EXTENDED OPTIONS

Longer names are used for options that are thought to be less commonly wanted, or are more likely to be used only within **--include** files.

--include *filename*

-p*filename*

-p Load one or more options from file *filename*. The options "**--include** *filename*" and "**-p***filename*" are equivalent; note that one requires a space before *filename* and the other cannot have a space (**-p** alone has a related function, described below). **ries** will attempt to open the named file (which may be a simple filename or a path), or the given name with "`.ries`" appended. If either is found, **ries** will scan it for parameters and arguments separated by whitespace. Any control characters count as whitespace. Any '#' character that comes at the beginning of a line or immediately after blank space denotes a comment and the rest of the line will be ignored. For example, if there is a file "`hst.ries`" containing the following:

```
# hst.ries:  High School Trigonometry settings
--trig-argument-scale 1.74532925199433e-2    # pi/180
-NLleEv      # No log, ln, e, e^x or arbitrary roots
-Ox      # Only allow one 'x' on the left-hand-side
-x      # Show equation roots as "x = 123.456"
        #            rather than "x = T + 1.23e-4"
```

then giving the option "**-phst**" is equivalent to giving the options "**--trig-argument-scale 1.74532925199433e-2 -NLleEv -Ox -x**", in that order.

A parameter file may additionally invoke another parameter file with the **--include** option. When it encounters this option, **ries** will apply the options in the included file, then continue with the rest of the first file. These may be nested up to 25 levels deep. If a file includes itself recursively (either directly or indirectly) **ries** will exit with an error.

It is an error for **--include** or the end of an included file to come between an option and its arguments. For example, "`ries 1.2345 --eval-expression --include expressions.txt`" will produce an error regardless of the contents of "`expressions.txt`", because **--eval-expression** must be followed immediately by its argument(s). On the other hand, if "`expressions.txt`" contains the **--eval-expression** option, like so:

```
# expressions.txt:  Useful functions of one argument
--eval-expression
  xsr    # 1/(x^2)
  1xq-r  # 1/(1-sqrt(x))
  2xl1+^ # 2^(ln(x)+1)
```

then the command "`ries 1.2345 --include expressions.txt`" works, and shows the values of the three expressions where *x* is 1.2345.

If you have a file called "`.ries_profile`" or "`ries_profile.txt`" in your home directory, **ries** will load it as if you specified it with a **--include** at the very beginning of the parameters. If you have such a file and wish to prevent it from being used, give a bare **-p** (without a filename) at the very beginning of your **ries** options. If you wish to give some parameters and have `.ries_profile` loaded *after* your parameters, include **-p** again at the point where you want **ries** to use the profile. For example, if your `.ries_profile` contains "**--trig-argument-scale 1**" and you have a `hst.ries` with contents as shown above, then giving the optiions "**-phst -p**" will use all of the settings in `hst.ries` except the **--trig-argument-scale**.

Two more example profiles are the "Latin" and "Mathematica" settings files linked from the top of the main RIES webpage.

--absolute-roots

This is a synonym for the **-x** option, described above.

--algebraic-subexpressions

This is a synonym for the **-a** option, described above.

--any-exponents

This option cancels any restrictions on subexpressions used as an exponent, such as those set by the **--algebraic-subexpressions** and **--liouvillian-subexpressions** options.

--any-subexpressions

This option cancels any restrictions on subexpressions, as imposed by options such as **--algebraic-subexpressions**. This might be useful if you are using one of the class selectors like **-a** or **-c** as shorthand for all the restrictions of that particular class, and then re-enable a function like e^x using **-EE**.

--any-trig-args

This option cancels any restrictions on subexpressions used as an argument to a trigonometric function, such as those set by the **--algebraic-subexpressions** and **--liouvillian-subexpressions** options.

--canon-reduction *symbols*

Apply simple transformations in an effort to make all expression values fall in the range [1 ... 2). This option improves the efficiency of the **ries** algorithm (described in the "ALGORITHM" section below) by increasing the chances of two expressions forming a match. This allows it to use less memory and time to achieve any given amount of precision.

This option should be followed by one or more symbols which represent the operations **ries** will try to apply to expressions:

n Negate expressions when possible to make all values positive.

r Take the reciprocal when possible to make all expressions fall outside the range (-1 ... 1).

2 Multiply by 2 when possible to increase the magnitude of expressions in the range (-1 ... 1).

5 Divide by 2 (i.e. multiply by 0.5) when possible to decrease the magnitude of expressions that fall outside the range (-2 ... 2).

In these descriptions, the words *when possible* refer to the fact that **--canon-reduction** will respect any limits imposed by the symbolset options **-N**, **-O** and **-S**. So if you use the option **-On** together with **--canon-reduction n**, the negation operator will still be used only once per expression.

Although it makes **ries** more efficient, this option also causes the printed results to have greater complexity scores, and complexity scores will increase somewhat more erratically. **ries** will try to simplify its printed results by undoing **--canon-reduction** transformations on both sides of the equal sign. For example, `ries 2.50618 --canon-reduction nr25` might yield the result “$x^x/2$ = (1+9)/2”, which simplifies to “x^x = 1+9”. But when only one side has a “/2”, **ries** cannot fix it, so the same example gives an overly complex “1/(*pi*-*x*) = *pi*/2”.

--canon-simplify

When reporting a match, remove common factors or terms from both sides of the equation. This is the default.

--constructible-subexpressions

This is a synonym for the **-c** option, described above.

--derivative-margin *value*

Specify the limit to how small the derivative of any expression or subexpression containing *x* can be in relation to the expression’s value. By default this is 10^{-6}, so that an expression containing *x* is rejected if its value is more than a million times its derivative. For really large target values, this doesn’t work because the expression “*x*” (with a derivative of 1.0) would be rejected. So if the magnitude of your target is larger than 10^5, **ries** will set this limit correspondingly lower. For example, if your target value is 10^8, **ries** automatically sets a **--derivative-margin** value of 10^{-9}.

If you do not consider these defaults suitable, use this option to pick your own value. For example in the command `ries 12345 --derivative-margin 8e-5`, derivatives of expressions can be as small as 8×10^{-5} times the expression’s value. In calculating any possible answers, **ries** would allow “x^2” because the ratio between d/d*x* x^2 and x^2 is about 1.62e-4, which is big enough to surpass the margin. However, any answers involving “sqrt(*x*)” would be rejected because the ratio between d/d*x* sqrt(*x*) and sqrt(*x*) is only 4.05e-5:

expression	value	d/d*x*(expr.)	ratio
x^2	152399025	24690	1.62e-4
x	12345	1.0	8.1e-5
sqrt(*x*)	111.108	0.0045	4.05e-5

Indeed, the results of `ries 12345` include the equation “2(sqrt(*x*)-9) = $(2\ e)^{pi}$”, but with the option **--derivative-margin 8e-5** that answer is left out.

--explicit-multiply

Always use the “*” symbol when displaying results, rather than the default behavior of omitting “*” when multiplicaton can be implied by writing the multiplicands next to each other. This is useful if you need to copy **ries** output into a calculator, computer program or spreadsheet formula.

--integer-subexpressions
This is a synonym for the **-i** option, described above.

--match-all-digits

--mad Request that all reported matches should match all of the supplied digits. This is equivalent to adding a '5' to the end of your target value, along with a **--max-match-distance** value equal to the magnitude of this appended '5' digit. It also selects the **--x** option, unless the **--wide** option is also given.

For example, the command `ries 2.5063 --mad` is equivalent to `ries 2.50635 --max-match-distance 0.00005 -x`, and the first reported match is x^2+e = 9, which is true for x = 2.506335... Without **--mad** it reports 2 x = 5 and a few other answers that do not match all of the digits in 2.5063.

--max-equate-value *value*

--min-equate-value *value*
Specify the maximum and minimum values for the LHS and RHS of any reported equations. For example, the command `ries 2.50618` would normally give ''2 x = 5'' as the first solution; both sides of that equation are about 5. But the command `ries 2.50618 --max-equate-value 3` instead gives ''x-2 = 1/2'' as the first answer: this is an equivalent solution, but expressed as an equation in which both sides of the equal sign are less than 3. Similarly, `ries 2.50618 --min-equate-value 27` gives the answer ''$(e^x)^2 = e^5$''.

--max-match-distance *value*
Specify the maximum distance between your given target value T and the roots x of any reported equations. This sets a minimum level of accuracy, overriding the default, which is 1% of the size of your target value. For example, the command `ries 2.5063` will use a threshold that is 1% of 2.5063, or about 0.025. It gives as its first answer the equation $2x$ = 5, an equation whose root (solution) is 2.5. This differs from the target value 2.5063 by 0.0063. If you specify an initial threshold of 0.001 with the command `ries 2.5063 --max-match-distance 0.001`, then $2x$ = 5 is not reported because 0.0063 is bigger than your threshold 0.001; instead the first match will be ''x^2 = 2 *pi*'' (which comes within about 0.0003 of the target 2.5063).

Use a zero argument to specify that **ries** should only report an ''exact'' match, if any (and note that this ''exact'' match might be more complex than the obvious answer, because of roundoff errors; see UNEXPECTED BEHAVIOR and BUGS below). Note that this is different from **--min-match-distance 0**, which prints inexact matches and stops after the ''exact'' match.

Use a negative argument to specify a match threshold in proportion to your target value. For example, `--max-match-distance -0.001` specifies that the first match must be within 1 part in 1000 (or 1/10 of one percent) of the magnitude of the target.

If your choice of **--max-match-distance** is so stringent that the first match takes longer than 2 seconds, **ries** will display progress messages until a match is found. Use the **--no-slow-messages** option to suppress these.

There is also a **--min-match-distance** option (described below), which serves an entirely different purpose.

--max-matches *N*

-n*N* Limit the number of reported matches to a positive integer *N*. This is particularly useful with certain options (such as **--no-refinement**) that generate a lot of matches. The default *N* is 100.

--max-memory *size*

This option tells **ries** not to use more than the given amount of memory (size specified in bytes). This is particularly useful in combination with a high **-l** (search level) option. For example, if you typically have about 2 gigabytes of free memory on your machine, you could invoke **ries** with the option **--max-memory 1.0e9**, to ensure that it never uses more than 1 gigabyte of memory regardless of the search level.

ries also has an (experimental) feature that can automatically detect when your system is slowing down; see the **--memory-abort-threshold** option for details.

--max-trig-cycles *value*

Specify the maximum and minimum values for arguments to the trigonometric functions, in multiples of one half-period. The default is 1.0, which corresponds to an allowed range of *-pi* to *pi*. The option `--max-trig-cycles 4` would allow arguments from -4*pi* to 4*pi*. In combination with **--trig-argument-scale**, this argument is applied after the scaling, so that the number of ‘‘cycles’’, ‘‘periods’’ or ‘‘waves’’ remains the same. For example, if the value *pi*/180 is used for **--trig-argument-scale** and if **--max-trig-cycles** is set to 2, then the allowed range for arguments to the trigonometric functions is -360 to +360 (which is interpreted as being in units of degrees), corresponding to two full cycles of the trigonometric functions. This option does not affect the arctan2 function, which always returns results from *-pi* to *pi* (1 full cycle).

--memory-abort-threshold *N*

This option is used with the **-DM** option, and overrides the default slowness measurement after which **ries** will automatically exit. With the **-DM** option, **ries** measures how fast it is running, as compared to an estimate of how fast it ‘‘should’’ be running. If this ratio is greater than the *--memory-abort-threshold* for more than 3 of the past 10 measurements, **ries** will exit. The default *--memory-abort-threshold* is 2.0. The value must be at least 1.0, and values less than about 1.5 are unlikely to be of much use.

NOTE: **--memory-abort-threshold** is an experimental **ries** feature and is likely to change in future versions of **ries**.

--min-match-distance *value*

Specify the minimum distance between your given target value *T* and the roots *x* of any reported equations. This is useful for finding approximate formulas for constants that have a known, simple formula. For example, using the command `ries 3.141592653589 -x --min-match-distance 1e-8 -NSCT` one can discover the following approximate formulas for *pi* :

```
            x-3 = 1/6            for x = 3.16666666666667
            x-3 = 1/7            for x = 3.14285714285714
      ln(ln(x)) = 1/e^2          for x = 3.14219183392327
          x^2+1 = 4 e            for x = 3.14215329254258
          e^x+2 = 8 pi           for x = 3.14124898321672
        x/phi^2 = 1/5+1          for x = 3.14164078649987
         e^x-pi = 4*5            for x = 3.14163154625921
      e^(x^2)+1 = e^(pi^2)       for x = 3.14158442136535
           pi-x = 1/e^(4^2)      for x = 3.14159254105462
   sqrt(1+pi) x = e^3/pi         for x = 3.14159272240341
        x^2/e^3 = 1/sqrt(1+pi)   for x = 3.1415926879966
        9(x-pi) = 1/-(e^(4^2))   for x = 3.14159264108588
```

Among these results (after solving for *x*) are the ancient approximations 19/6 and 22/7, and the more modern curiosity $e^{pi} \cong 20 + pi$ (which is called "*Gelfond's constant*"). Other interesting results can be found by omitting symbols with **-N** or by using restricted classes. For example **ries 3.1415926 -NASCTlLfEevp --min-match-distance 1e-8** (excluding most of the scientific functions) gives the fraction approximation 355/113 in the form "1/(*x*-3)-1 = 1/4^2+6"; and **ries 3.1415926 -r --min-match-distance 1e-8** (requesting only rational approximations) gives 355/113 in the form "1/(*x*-3)-3 = 1/(4*4)+4".

You will often get multiple equivalent results. In the above example, the equations sqrt(1+*pi*) *x* = e^3/*pi* and x^2/e^3 = 1/sqrt(1+*pi*) can both be converted into the approximate relation:

$$pi \cong \mathrm{sqrt}(\mathrm{sqrt}(e^6/(pi+1)))$$

(which does *not* converge on the true value of *pi* if iterated).

If you give a value of 0: "`--min-match-distance 0`", and **ries** finds an "exact" match, it will exit and report no further results. Note that this is different from **--max-match-distance 0**, which will only print the "exact" match and will not print any inexact matches.

There is also a **--max-match-distance** option (described above), which serves an entirely different purpose.

--min-memory *size*

If **ries** is given the debug option **-DM**, it will try to measure the responsiveness of the system and automatically exit if it gets very slow. This is intended as an automatic safeguard against virtual memory "thrashing" that will happen if **ries** is allowed to use all of your system's memory. (This feature is only active with the **-DM** option because it is still being tested).

When **--min-memory** is given in combination with **-DM**, it will ensure that **ries** does not exit because of slow memory response until at least *size* bytes of memory have been used. For example, if you know that you always have about 1 gigabyte of free memory on your machine, and your machine often gets slow for other compute-intensive tasks, you could invoke **ries** with the options **-DM --min-memory 1.0e9**, and slow system detection would be enabled but would not trigger (if at all) until a gigabyte of memory has been used.

For more direct control over **ries**' memory usage, use the **--max-memory** option (without **-DM**) or use a suitably small **-l** search level.

--no-canon-simplify

When reporting a match, do *not* remove common factors or terms from both sides of the equation. This is useful mainly in combination with **--max-equate-value** and **--min-equate-value**. For example, the command `ries -Ox 2.6905` would normally give the answer "*x*-2 = ln(2)" in which both sides of the equation are about 0.693. Adding the options **--no-canon-simplify** and **--min-equate-value 1** reports the same answer as "1/(*x*-2) = 1/ln(2)" in which both sides of the equation are about 1.443.

--no-refinement

After reporting a match, do *not* require that the next match come closer to the target. This causes **ries** to emit many more matches than it normally would. The matches will not be given in order of closeness, but they will still be (roughly) ordered by increasing "complexity". Many will be equivalent to one another, for example the command `ries 1.51301 --no-refinement` yields the solutions e^(x^2) = pi^2 and *x*/sqrt(2) = sqrt(ln(pi)), both with the root 1.513096088... This option is most effective in combination with **--max-matches** along with **--match-all-digits** or **--max-match-distance** (using a stricter argument than the default -0.01).

--no-slow-messages

Suppress the ‘‘Still searching’’ messages that **ries** would normally print if a search takes longer than 2 seconds without giving any results.

--no-solve-for-x

This option cancels the ‘‘**--try-solve-for-x**’’ option.

--numeric-anagram *digits*

Give a specific set of digits that can be used as constants in a solution; this forces the **--one-sided** option. It will use only as many of each digit as you specify. For example, if you give ‘‘111223’’ as the digits, **ries** will use up to three 1’s, two 2’s, and/or a single 3. This is meant to aid in solving certain puzzles of the ‘‘four fours’’ variety:

```
ries 17 -ie --numeric-anagram 4444
     x = 4*4+4/4                ('exact' match)       {92}

ries 17 -ie --numeric-anagram 111223
     x = 2^(1+3)+1              ('exact' match)       {77}

ries 12 --numeric-anagram 44s
     x = 4^2-4

ries 12 --numeric-anagram 442
     x = 2*4+4
```

--numeric-anagram can be used to set hard limits on the digits, the constants *e*, *phi*, and *pi*, and the ‘‘squared’’ and ‘‘reciprocal’’ symbols. When you use **--numeric-anagram**, any of these symbols that you do not list will be forbidden just as if you had used the **-N** option. The last two examples here show the use of ‘‘s’’ to specify the squaring operation x^2 as distinguished from any use of ‘‘2’’.

--one-sided

Force **ries** to ignore all LHS expressions except *x*. This results in ‘‘one-sided equations’’ with *x* on the left-hand side. This makes **ries** much slower, but all of its output will be ‘‘solved for *x*’’.

This option is intended as a convenience for very special problems (for example, **--numeric-anagram** automatically turns on **--one-sided**), but it is not generally useful because the speed advantage of the normal RIES bidirectional search algorithm is lost. If you want **ries** to give answers that are solved for *x*, use the **--try-solve-for-x** option possibly along with **-Ox**.

If you use this option, **ries** will be a lot slower and its solutions for a given search level will not be nearly as accurate. Whereas a normal **ries** search might quickly match your target value to the first 10 decimal places, a search with the **--one-sided** option, taking the same amount of computation time, will only match the first 5 decimal places. Sometimes this is acceptable, particularly when used with other options that restrict the search, such as **-i**, **-N**, **-O**, and **-S**.

--rational-exponents

Require any exponent to be a rational subexpression. For example sqrt(2) is allowed because it is 2 to the power of 1/2, but 2 to the power of sqrt(2) is not allowed because sqrt(2) is not rational. This option exists mainly to support the **-a** or **--algebraic-subexpressions** option (described above).

--rational-subexpressions

This is a synonym for the **-r** option, described above.

--rational-trig-args

Require any argument to the forward trigonometric functions (sine, cosine, tangent) to be a rational subexpression. The restriction applies to the argument's value before multiplying by the **--trig-argument-scale** (if any). This option exists mainly to support the **-a** or **--algebraic-subexpressions** option (described above).

--relative-roots

When printing each equation, show the root as *T* plus/minus a small number (where *T* is your target number) rather than as the actual value of the root. This is the default, so you'll only need to use **--relative-roots** to cancel a **-x** or **--absolute-roots** option in your `.ries_profile` or another **--include** file.

--ries-arguments-end

This 'option' signals the end of options and arguments; any that come after it will be ignored. If it occurs within a profile (described under the **--include** option above) **ries** will ignore the rest of the contents of that file and continue with the next option after the **--include** or **-p** option that invoked the profile.

--show-work

This is a synonym for the **-Ds** option, described in the **-D** (debugging/diagnostic) option above.

--significance-loss-margin *digits*

Specify the number of significant digits that may be lost in a calculation. By default, **ries** tolerates a loss of 2 digits in any calculation. For example, if x is 0.906402477... (the value of Gamma[5/4]), **ries** would not use x+e^5 in any of its expressions, because e^5 is more than 100 times as large as x. Due to round-off, more than 2 digits of the value of x would be lost in the addition. This restriction applies to constant expressions too, so **ries** also avoids 1+e^5 and 1+e^{-5}. Similar restrictions apply to any function that can cause precision to be lost (if evaluated at a point where the function's derivative is very low).

The normal **ries** behavior corresponds to a **--significance-loss-margin** option with an argument of 2.0. Give a higher value to allow more digits to be lost in calculations.

Conversely, if you suspect **ries** is generating meaningless results due to round-off error, you can look at its calculations in detail with the options **-Ds** and **-F0**, then evaluate specific expressions with **--eval-expression** (described below). If it seems appropriate, make **ries** more strict by giving **--significance-loss-margin** with a lesser argument.

--symbol-names :*sym*:*name* [:*sym*:*name* ...]

This option allows you to set the "names" of individual symbols. This affects how the expressions and equations are printed in any of the **-F** formatting modes, and by special commands such as **--eval-expression**. In addition to the symbols listed above in the **-N** option, you may also define these symbols:

( ) brackets to group sub-expressions

= equality symbol

Here are examples of the normal **ries** output, modified by changing the appearance of the exponentiation operator and parentheses. The single-quotes around each option are to avoid substitution by the shell:

```
ries -l0 2.5063
                 2 x = 5                      for x = T - 0.0063      {49}
                 8 x = e^3                    for x = T + 0.00439212  {66}
                 x^2 = 2 pi                   for x = T + 0.000328275 {55}
                 x^x = 1+9                    for x = T - 0.000115854 {69}
               x^2+e = 9                      for x = T + 3.56063e-05 {63}
             ln(6) x = sqrt(pi)+e             for x = T + 2.73037e-05 {93}

ries -l0 2.5063 --symbol-names ':^:**' ':(:[' ':):]'
                 2 x = 5                      for x = T - 0.0063      {49}
                 8 x = e**3                   for x = T + 0.00439212  {66}
                x**2 = 2 pi                   for x = T + 0.000328275 {55}
                x**x = 1+9                    for x = T - 0.000115854 {69}
              x**2+e = 9                      for x = T + 3.56063e-05 {63}
             ln[6] x = sqrt[pi]+e             for x = T + 2.73037e-05 {93}
```

More examples of the use of **--symbol-names** are found in the ‘‘Latin’’ and ‘‘Mathematica’’ settings files linked from the top of the main RIES webpage.

--symbol-weights *N*:*sym* [*N*:*sym* ...]

This option allows you to adjust the ‘‘weights’’, or complexity ratings, of individual symbols. Use the option **-S** to see the normal weights, then use this option to change one or more. Compare these two examples; in the second one the cost of the symbol *x* is reduced, and the costs of **2** and **s** (squared) are increased.

```
ries -l0 2.5063
                 2 x = 5                      for x = T - 0.0063      {49}
                 8 x = e^3                    for x = T + 0.00439212  {66}
                 x^2 = 2 pi                   for x = T + 0.000328275 {55}
                 x^x = 1+9                    for x = T - 0.000115854 {69}
               x^2+e = 9                      for x = T + 3.56063e-05 {63}

ries -l0 2.5063 --symbol-weights 5:x 25:2 15:s
           x sqrt(x) = 4                      for x = T + 0.0135421   {39}
                -x-x = -5                     for x = T - 0.0063      {46}
               x/x^x = 1/4                    for x = T + 0.00150933  {49}
               x x^x = 5^2                    for x = T - 0.00118185  {57}
                 x^x = 1+9                    for x = T - 0.000115854 {49}
           x (1/x-x) = e-8                    for x = T + 3.56063e-05 {71}
```

With the smaller weight of 5, *x* is considered ‘‘less expensive’’, and **ries** uses *x* more often in its answers; and with the number **2** and squaring more expensive, these show up less often in the results. In many cases the new results are equivalent, and **ries** has simply found a different way to get there.

Since the arguments of **--symbol-weights** start with a digit, your target value will be treated as a symbol-weight specifier unless you place it somewhere else in the parameter list (as shown in the example), or use a single dash ‘‘-’’ to signal the end of the parameters.

NOTE: Changing the symbol weights can greatly reduce **ries**’s efficiency, causing it to run for a very long time and giving little or no output. If this happens, it usually can be fixed by using weights closer to the default values. You can also experiment with changing just one symbol-weight at a time to find which is causing the problem.

--trig-argument-scale *value*

Specify a constant by which the argument of the sine, cosine and tangent functions should be multiplied. By default this value is *pi* and the trig functions are called `sinpi`, `cospi` and `tanpi`. sinpi(*x*) is the sine of *pi* times *x*; so for example sinpi(1/3) is the sine of *pi*/3, which is half the square root of 3. A full circle is 2 in these units: sinpi(*x*) = -sinpi(*x*+1) = sinpi(*x*+2) for all *x*.

If you give this option, arguments will be multiplied by the number you give instead of by *pi*. Useful values to give are:

6.2831853071795864769
This is "tau" (2 *pi*); use it to get units of 1 per "full turn": "sin(1/16)" will give 0.382683...

1 Use 1 to get natural units (radians): "sin(pi/8)" will give 0.382683...

1.74532925199432957692e-2
This is *pi*/180, and is used for degrees: "sin(22.5)" will give 0.382683...

1.57079632679489661923e-2
This is *pi*/200, and is used for grads (gradians or gons): "sin(25)" will give 0.382683...

If you use this option, **ries** will call the functions `sin`, `cos` and `tan` in its output, and the scale will be displayed after the function definitions at the end.

--try-solve-for-x

-s **--try-solve-for-x** is equivalent to "**-s**", which is described in the OPTIONS section above.

--version

Displays information about the version of **ries**, the calculation precision and math library, any optional module(s), the currently enabled profile (see the **--include** option at the beginning of the EXTENDED OPTIONS section above), and a brief copyright notice. The version is a date, such as "2013 Jun 3".

--wide-output

Use a wider (132-column) output format. This shows the roots of equations (values of *x*) both in terms of the the actual value of *x*, and as *T* plus/minus a delta; it also shows the ratio between this delta and *x* as "(1 part in *N*)" where *N* is the delta divided by *x*. For example, if *x* is 2.5 and target value is 2.501, the delta is 0.001, which is "1 part in 2500".

SPECIAL COMMANDS

ries provides some functions that supplement its main purpose. Commands and their parameters must be separate: '`ries 1.23 --trig-argument-scale 0.5`', not '`ries 1.23 --trig-argument-scale 0.5`'. Because the parameters are given separately, your target value might be interpreted as a parameter if you give it right after a special command. To avoid this, use a single dash "-" to signal the end of the parameters.

--eval-expression *forth-expr* [*forth-expr* ...]

Evaluate one or more expressions, showing intermediate values, derivatives, and the complexity score of the full expression. The expression(s) should be given in the FORTH-like postfix syntax that is displayed when you use the -F0 option. The symbols are as listed above under the **-N** option. For example, `xxq-` is the syntax for *x*-sqrt(*x*). Syntax errors and computation errors such as overflow are reported; however successful execution by **--eval-expression** does not guarantee that the expression will be found in an actial **ries** search. For example, an expression containing *x* only appears as the left-hand-side of an equation if the *x* is the first symbol in the postfix form: **ries** will use `x2+` (*x*+2) but will not use `2x+` (2+*x*).

--find-expression *forth-expr* [*forth-expr* ...]

Perform the normal equation-finding search algorithm, and report specific expressions when they are found, along with their value, derivative, and complexity. The expression(s) should be given in the FORTH-like postfix syntax that is displayed when you use the -F0 option. The symbols are as listed above under the **-N** option. For example, `xxq-` is the syntax for *x*-sqrt(*x*). This command is useful for diagnostics; an example is given below in the UNEXPECTED BEHAVIOR section.

STAND-ALONE MATHS LIBRARY

The **ries** source code includes an auxiliary file, `msal_math64.c`, which can be downloaded from the same place as the main RIES source code and this manual. It provides some of the standard trignonmetric functions, whose implementation have been found to vary across different releases of the standard `libm`. This is useful if you are running **ries** on a variety of newer or older systems and want to be able to rely on consistent results.

`msal_math64.c` also provides the Lambert W function, defined to the symbol ’W’.

To use `msal_math64.c`, compile **ries** in the normal way but with the additional compiler option `-DRIES_USE_SA_M64`. The resulting **ries** binary will report ‘‘mathlib: stand-alone’’ when given the `--version` option.

Once **ries** has been compiled with the stand-alone maths library, the Lambert W function is available by giving the option `-EW` on the **ries** command-line. Its default weight is set to make it occur a little more often than the two-argument exponential and root functions; use the **--symbol-weights** option if you want to change this.

ALGORITHM

ries begins its search with small, simple equations and proceeds to longer, more complex ones. It uses a set of *complexity rules* to compute a measure (similar to Kolmogorov complexity), which determines the order in which various candidate expressions are considered by **ries**. For example:

1. If you add a symbol to an equation, the result is more complex:

   $x + 1 = 3$ is more complex than $x = 3$

   $x + 1 = \ln(3)$ is more complex than $x + 1 = 3$

   $x - 7 = 4\hat{}2$ is more complex than $x - 7 = 4$

2. If two equations are the same except for one number, the equation with the higher number is more complex:

$x + 1 = 5$ is more complex than $x + 1 = 3$

x^3 + 1 = 3 is more complex than x^2 + 1 = 3

3. If two equations are the same except for one symbol, the equation with the "more exotic" symbol is more complex:

x ^ 5 = 3 is more complex than $x + 5 = 3$

As **ries** searches it finds solutions -- these are equations for which x is close to being an exact answer. Each time it finds a solution it prints it out. Then **ries** raises its standard for the next answer: The next answer **ries** prints must be a closer match to your supplied value than all the answers it has given so far. (The only exception to this rule is an 'exact' match, one for which both sides match to within the limits of numerical precision. **ries** will print at most one of these, and will then continue to print more inexact solutions. Sometimes the approximations are of greater interest than the exact match.)

Instead of trying complete equations, **ries** actually constructs half-equations, called *left-hand-side expressions* and *right-hand-side expressions*, abbreviated LHS's and RHS's. It keeps a list of LHS's and a list of RHS's, and it keeps these lists in numerical order at all times. This enables **ries** to find matches much faster. All LHS's contain x and all RHS's do not. Thus, 1000 LHS's and 1000 RHS's make a total of 1000000 possible equations, and all 1000000 combinations can be quickly checked just by scanning through the two lists in numerical order. This is why **ries** is able to check billions of equations in such a short time.

The closeness of an LHS match depends on the value of x, and also on the derivative with respect to x of the LHS expression. Because of this, **ries** calculates derivatives of LHS's as well as their values.

There are dozens of optimization rules **ries** uses, like the following:

a+ Don't try "K + K" for any constant **K** because "K * 2" is equivalent.

b+ Don't try "3 + 4" (or any two unequal integers from 1 to 5) because another single integer (in this case "7") is shorter.

a* Don't try "1 * K" for any constant **K** because "K" is shorter.

b* Don't try both "2 * 4" and "4 * 2" because they are equivalent.

c* Don't try "K * K" because "K ^ 2" is shorter.

ar Don't try "1 / (1 / expr))" for any expression **expr** because "expr" is shorter.

a^ Don't try "2 ^ 2" or "2 squared" because "4" is shorter.

b^ Don't try "expr ^ 2" for any expression **expr** because "expr squared" is shorter.

There are over 50 rules like this, and together they make the search about 10 times faster. However, if the symbol set is limited via **-N**, **-O** or **-S**, some of these rules cannot be used. For each optimization rule there are one or more symbolset exclude rules like the following:

Don't use rule **a+** if either of the symbols '*' or '2' is disabled.

In order to maintain maximum efficiency, **ries** checks each rule individually against the symbolset, and uses as many rules as it can. You can see this process in action by trying a command like `ries 1.4142136`, which gives the answer "$x^2 = 2$". If you disable the 's' (squared) and '^' (power) symbols with `ries`

`1.4142136 -Ns^`, rule **b^** goes away, and the answer becomes "*x x* = 2". If you also disable '*' (multiplication) the answer becomes "*x* = sqrt(2)". Disable 'q' (square root) and you get "log_2(*x*) = 1/2" (the logarithm to base 2 of *x* is 1/2). Disable 'L' and it becomes "2/*x* = 2,/2" (2/x is the square root of 2, this time using the generalized root function). Disable 'v' and you get "*x*/(1/*x*) = 2". These are all trivially equivalent to the original "x^2 = 2", but as the easier operations like multiplication are removed, exclude rules go away making it possible to get to the same answer in an indirect way.

Continuing, we may disable '/' to get a trignometric identity involving *pi*/4; disabling the trig functions as well, the command becomes `ries 1.4142136 '-Ns^q*Lv/ASCT'` and we finally get an answer that most people probably would not guess: *x*-1/(*x*+1) = 1 (note that the '/' in this answer is actually part of '1/', which is the reciprocal operator 'r'). Throughout this progression the complexity score of the equation generally increases as the solution becomes more and more obscure, and simpler but poorly matching approximations like 1/cos(*x*) = 6 begin to appear before the correct one.

UNEXPECTED BEHAVIOR

Sometimes a more complex equation will be given before the (simpler) equation that you expect. For example, tan(sqrt(2)) = sin(sqrt(2))/cos(sqrt(2)) is 6.334119167042..., so you might expect the command `ries 6.334119167042 --trig-argument-scale 1 -NT` to report something like "cos(sqrt(2)) *x* = sin(sqrt(2))". Instead, **ries** gives "sqrt(x^2+1) = 1/cos(sqrt(2))", which is equivalent by trigonometric identity, because it considers this equation to be "more balanced" (complexity score 45+38) than the other (which has a score of 48+29). Please read the preceding section "ALGORITHM" for more details.

Adding or changing the symbolset with the **-S**, **-O** and **-N** options often causes unexpected changes in the output. For example, `ries 2.2772` yields the solution "1/(*x*-2) = 2+*phi*" but `ries 2.2772 '-N*/T'` does not give this solution in any form. This seems counterintuitive: there was no *, /, or tan() in the "1/(*x*-2) = 2+*phi*" solution, so why did **ries** decide not to report it?

In fact, the solution is still generated internally, but because you have told it to exclude some operators, **ries** has to try other, more exotic expressions sooner than it otherwise would. As it happens, the next solution "*x*+1/e = sqrt(7)" (which matches the target value 2.2772 more closely than "1/(*x*-2) = 2+*phi*") ends up getting found earlier.

The mysterious behavior results from the fact that **ries** always tries to keep the number of LHS and RHS expressions equal as it performs its search. Eliminating operators with the **-N** option means that more complex expressions must be generated to reach the "quota". In this particular case, the symbolset restriction has a greater effect on the LHS than on the RHS, so as the search is progressing, LHS complexity grows a little more quickly than RHS complexity. The complexity of "1/(*x*-2)" is 40, while the complexity of "*x*+1/e" is only a little more complex at 42. But the right-hand-side "sqrt(7)" is considered less complex (27) than "2+*phi*" (35). Since both pieces of an equation need to be found before an equation can be reported, **ries** is able to locate both pieces of "*x*+1/e = sqrt(7)" sooner when the `-N` option is given.

This is all made plain with the **--find-expression** option, giving the expressions "1/(*x*-2)", "2+*phi*", "*x*+1/e" and "sqrt(7)" in postfix form, to reveal the order in which they are generated by the search algorithm:

```
ries 2.2772 --find-expression x2-r f2+ xer+ 7q
 [7q] = 2.64575131106459, complexity = {27}
 [x2-r] = 3.60750360750361, d/dx = -13.0141, complexity = {40}
 [f2+] = 3.61803398874989, complexity = {35}
 [xer+] = 2.64507944117144, d/dx = 1, complexity = {42}
```

Then, repeating the same command with **'-N*/T'** shows that the `[xer+]` is generated before `[f2+]`.

In the case of two equivalent solutions (like the 6.334119167042... example earlier in this section), both equations come equally close to the supplied value, but only one can be found first. Once the first one is reported, the other is not, because **ries** only reports solutions that are at least a 0.1% closer match than the previously-reported solution.

The **-l** option is meant to give control over the number of solutions searched, but it actually controls the number of LHS and RHS expressions generated. Because two RHS's often have the same value, and only one (the first) gets kept, the number of solutions checked (which is the RHS count times the LHS count) depends on how often you get two LHS's or RHS's with the same value. This happens particularly often when the symbol set is severely restricted. If **ries** tried to compensate for this, the result would be that severely limited symbolsets would take a very long time to run and would generate really long equations. This is an important issue for those using **ries** to solve special problems, like the "four fours problem" exemplified by the command `ries --numeric-anagram 4444 -Ox 17`. The current implementation represents the author's attempt at a reasonable tradeoff.

BUGS

Performance does not degrade gracefully when the physical memory limit is hit, because expression nodes are allocated in sequential order in memory, without regard to where they will end up in the tree. This could be improved in the future with percentile demographics and a sort performed one time only, after the tree reaches a healthy (but not excessive) size.

Although it tries to avoid it, **ries** will often print more than one equivalent solution. It misses the fact that the multiple solutions are equivalent because of roundoff error. For example, `ries '-S4+-*/' -Ox 17` gives "x-4*4 = 4/4" and "(4/(4-4*4)) x+4 = (4*4+4)/(4-4*4)" (or a similar long form), only recognizing the first as an exact match (the more complex one involves divisions by multiples of 3, which require rounding).

This problem is common when the target is already known very precisely by the user. For example, `ries 0.00088953230706449` gives the (correct) answer "ln(x)/pi = -sqrt(5)", followed by redundant/equivalent answers such as "ln(x)/pi-2 = -(phi^3)" (if **ries** was compiled with regular precision) or "sqrt(5),/x+4 = 1/e$^{\text{pi}}$+4" (if compiled with the `-DRIES_WANT_LDBL` option); but `ries 0.000889532307` only gives the first, simplest form of the answer.

Related to this, **ries** sometimes gives an overly-complex answer, again because of roundoff error. For example, **ries** gets slightly different values for "2/3" and "1-1/3", and stores both of these in its database of RHS values. When reporting a solution in which both sides of the equation equal 2/3, it might give "1-1/3" for the right-hand side if it is closer to the (rounded) value of the left-hand side.

This is particularly common if you request an exact match with a zero or very small **--max-match-distance** value, while giving an imprecise target value. For example, `ries 1.9739208802178 --max-match-distance 0` might give "cospi(1/(5 x)^2) = cospi(1/pi^4)" whereas `ries 1.9739208802178715 --max-match-distance 0` gives the expected "5 x = pi^2". However, I've only gotten complaints about this from users who give **ries** a problem to which they already know an answer.

If you know that 1.9739208802178... is $pi^2/5$, then you don't really need **ries** to tell you that, do you? And if you're searching for things that approximate $pi^2/5$, such as sqrt(sqrt(e^2+1)+1) = 1.9739267..., you can use the **--min-match-distance** option.

In deeper searches or with target values larger than 10^5, **ries** might occasionally report "solutions" that are actually tautologies empty in meaning. A typical example is "$x^{(4/\ln(\text{sqrt}(x)))}$ = sqrt(e)$^{(4\hat{}2)}$" (which is true for any value of *x*), but **ries** handles that case and most others like it. If you suspect the solutions it gives, use the **-Ds** option to show all calculations behind each proposed solution. The options **--derivative-margin**, **--min-match-distance**, and **--significance-loss-margin** may help avoid meaningless results.

You can also use **-Ox** to force **ries** to use only a single *x* in each equation, which will prevent these tautologies entirely, but will also prevent the discovery of interesting solutions like $x^x = 10$.

ACRONYM

ries (pronounced "reese" or "reeze") is an acronym for "RILYBOT Inverse Equation Solver". The expansion of *RILYBOT* includes two more acronyms whose combined length is greater than 11. The full expansion of **ries** grows without limit and is well-defined but not primitive-recursive. Contact the author for more information.

AUTHOR

Robert P. Munafo (contact information on mrob.com)

LICENSES