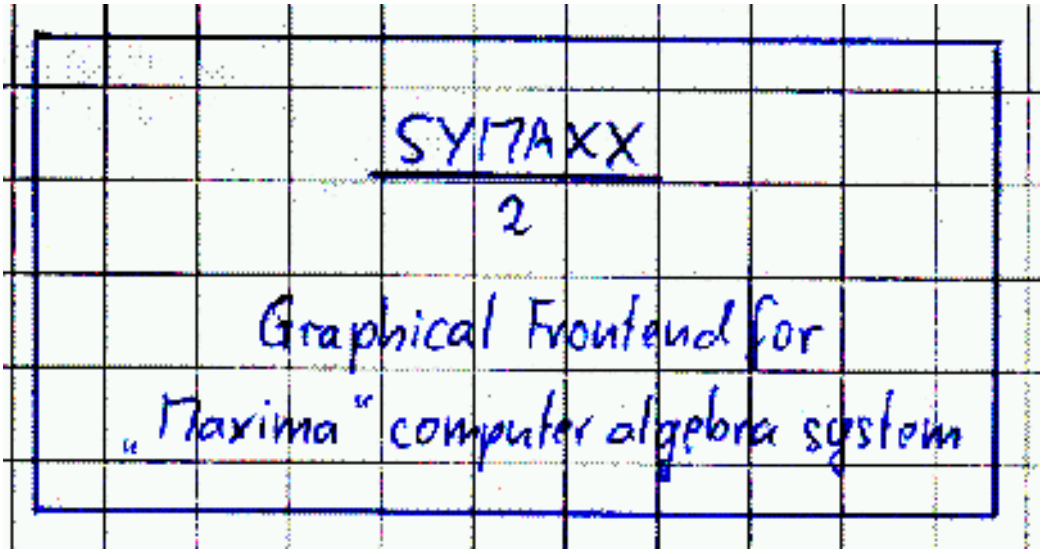

Manual, Release 0.5

Markus Nentwig

Please check also the homepage at http://symaxx.sourceforge.net. It may be more up-to-date than this manual.

1 What is Symaxx/2?

Symaxx/2[1] is a graphical frontend for the 'Maxima' computer algebra system. It has no mathematical 'intelligence' of its own, all the work is done by Maxima.

2 How does it work?

Figure 2 shows an example for a Symaxx object. It consists of

- at least one command line: $\int_1^2 \sin(x)\mathrm{d}\,x$
- at least one result line: $\cos(1) - \cos(2)$

[1] It is a complete rewrite of 'Symaxx'.

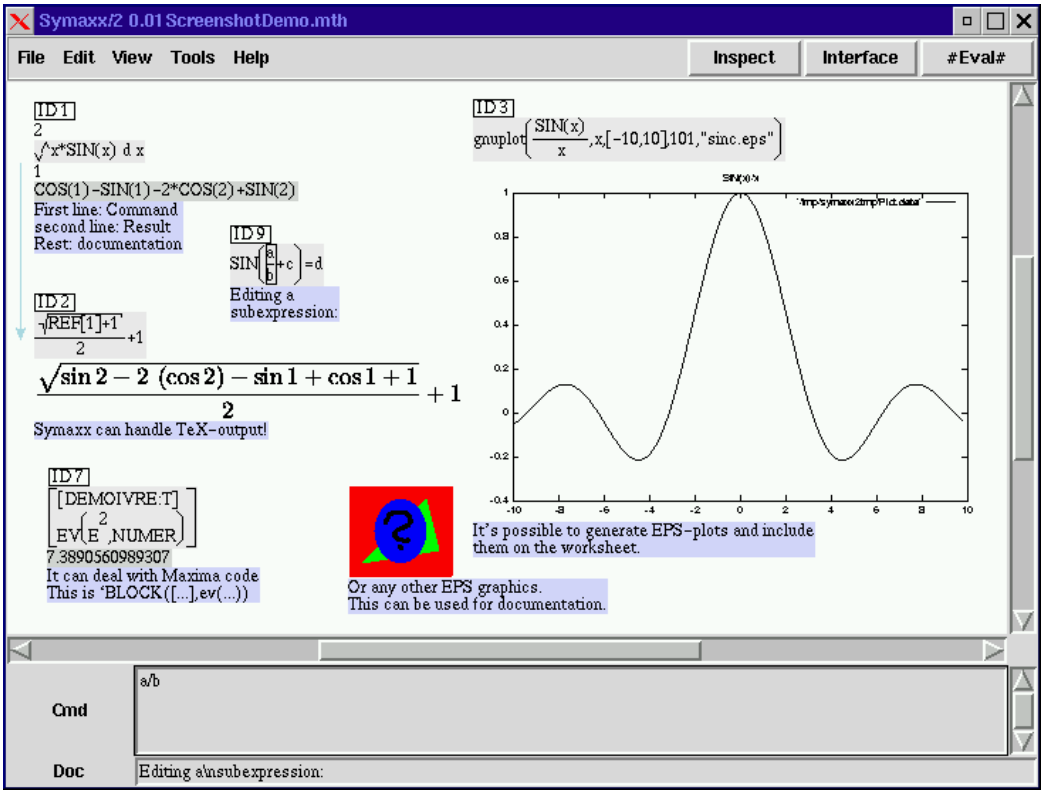

Figure 1: Screenshot

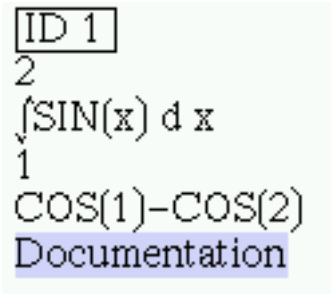

Figure 2: Example for Object

- A field containing optional text for documentation
- The header with a unique identifier: ID 1

- Flags that indicate the status (here none visible)
- Some properties: For example, you can turn on or off any of the components on the display.

2.1 The editor

To create a new object, left-click into a clear space (to deselect a current selection) and press the middle mouse button to create a new object at the current mouse position.

Click on the command (which is the upper '1'). The contents of the Cmd-line at the lower end of the screen change to '1'. Enter any Maxima command (3+4). Press the RETURN key. The command changes. If you click on something else instead of pressing RETURN, your changes are discarded.

You can select most subexpressions and edit them independently.

Usually you *don't* want to edit the result (the lower line).

Symaxx receives its input data in Maxima's internal format (Lisp-lists). The editor knows many commands and will display them in a convenient way.

But Symaxx does not yet include a graphical representation for all Maxima-commands. If an unknown command is used, Symaxx shows the command in the way Maxima stores it (as a function with one or more arguments). Figure 3 shows an example.

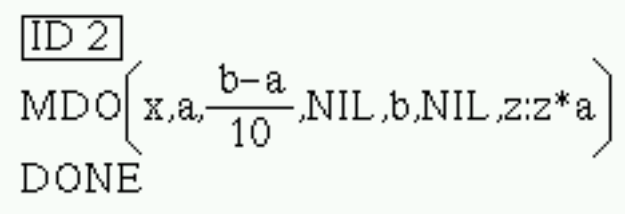

Figure 3: A command unknown to Symaxx (For-loop)

Figure 4 shows, what you will see in the editor, if you select the whole 'MDO'-command.

Even if the graphical representation differs from the standard ASCII form, you can still edit it using the normal Maxima syntax. And you can pick out a single argument and edit it alone - for example $\frac{b-a}{10}$ or $b - a$.

However, many commands *are* known to Symaxx already (figure 5).

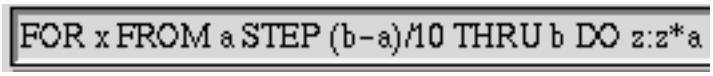

Figure 4: The same command as in figure 3 in the editor line

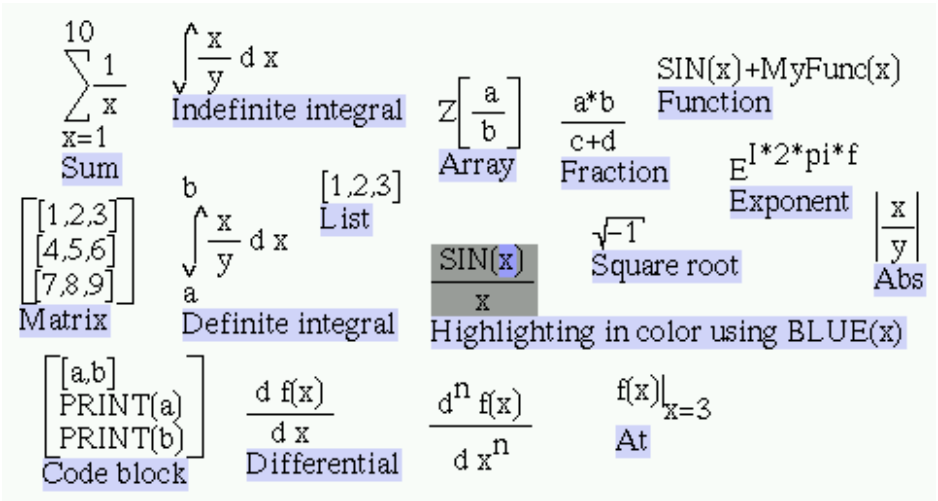

Figure 5: Functions with associated graphics

2.2 Command

The upper line (below ID x) is the command that you want Maxima to evaluate. Type any valid Maxima-command. Just entering the command will not evaluate it. For example, putting $3 + 4$ here will result in $3 + 4$ appearing in the command line.

A shortcut: Ending a command with $ (for example 3+4$) will turn off the display for the result. Entering a ; (3+4;) will turn it on again.[2]

2.3 Result

As soon as you run a command through Maxima (by hitting the #Eval# button or CTRL-Space), all command lines are evaluated and the results updated. For the $3 + 4$-command, this will result in 7 in the results-field.

[2]The $ or ; is stripped away at the moment the return-key is pressed. It is not stored. Instead, a flag is set / cleared.

You can (but should not) edit the result instead of the command. Since life often involves a lot of trial-and-error, it may be useful to 'override' a result sometimes. If you do this, a flag is set (see section 2.9, the background turns red). This result will NOT be updated anymore, if you hit the #Eval#-button. To turn the behaviour back to normal, edit the previous command[3] or clear the flag in the 'Properties'-dialog (see section 2.11).

2.4 Second / third / ... command

An object can contain any number of commands and results.

To add a new command, select the whole (!) result line and press the middle mouse button or Alt-n. Refer to the previous result by using % (i.e. %/2+1). Also, the %TH() function allows to access earlier results (i.e. %TH(2) - the result before %)[4]

2.5 Modifying parts of a result[5]

One of the most powerful, versatile (but also 'dangerous') capabilities of Symaxx is to manipulate subexpressions. For example[6], assume you want to replace the x in $\sin(x)$ with $\frac{x}{2}$.

First enter a command $\sin(x)$. Evaluate it. Click on the x in the result and press Alt-n or the middle mouse button. A new command appears, connected with a line to the 'x' (figure 6). This is called a *modifier*. The modifier '%' will replace the 'x' by its previous value (which is again 'x' - it does nothing - yet).

Click on the modifier and replace it with $\frac{\%}{2}$. Evaluate again (figure 7.

You will note, that a new command line has been inserted at the same instant, when the new modifier was created.

The internal logic of Symaxx works as follows:

- Sort all modifiers according to 'depth'

[3] You don't have to change it. Just select the command and press RETURN in the entry line.

[4] The %TH function is emulated by Symaxx.

[5] Note: This is an 'advanced' feature, useful but not essential. In fact, if you can avoid to use it, so much the better.

[6] Obviously, AT(sin(x),x=2*x) will produce the same result for this simple example.

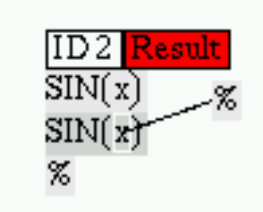

Figure 6: Manipulating a subexpression

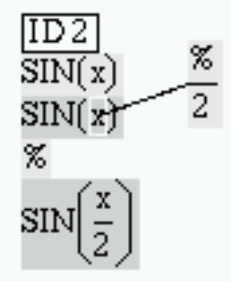

Figure 7: Manipulating subexpressions II

- Apply all modifiers, start with the innermost modifier (only substitution, no evaluation)
- Apply the next command (which is treated as top-level modifier)
- Evaluate

I suggest to use this feature only, if there is no easier way to get the same result using RATSUBST, AT or PICKAPART.

Why dangerous? Because if the structure of the original expression changes, your programmed manipulations will most probably not do what you expected them to. In the best case there will be an error message saying 'INPART fell off end'. In the worst case, you'll have an error in the calculation, that's hard to find.

2.6 Documentation

2.6.1 Standard documentation line

Each object has one line at its bottom, which serves only for documentation purposes. If the screen gets cluttered by too much documentation, it may help to turn off the doc-display for some objects[7] (see section 2.10). Linebreaks can be inserted with CTRL-Return.

2.6.2 Documentation within a Maxima expression

Longer documentation can be included as a normal mathematical object:

EPS pictures can be included by writing *EPS(“filename”,Scale)*. “Scale” will usually be “1”.

Please note, that Maxima does not recognize the “EPS” function. Therefore, Maxima evaulation does not change an EPS statement, as long as the arguments cannot be simplified (string, number).

Similar functions are provided to write text blocks and headings: Those are

- TEXT(“Text”)
- HEADING1(“Text”)
- HEADING2(“Text”)
- HEADING3(“Text”)

“Text” must be a string or number. Anything else will not be accepted.

The contents of a TEXT() statement are line-wrapped in Symaxx.

To separate several paragraphs, use YLIST(TEXT(“...”),TEXT(“...”),...)

[7] The documentation will be stored, so you can turn it on later.

2.7 Header and unique identifier

If you want to refer to the result of one of your calculations later, you need the ID number of the result. For the example given in figure 2, you could refer to the result of the integration by writing REF[1]. Note the capital letters and the square brackets! (REF is an array).

For example, REF[1]+1 results in $1 + \cos(1) - \cos(2)$.

As shown in figure 8, the dependency is represented by an arrow.

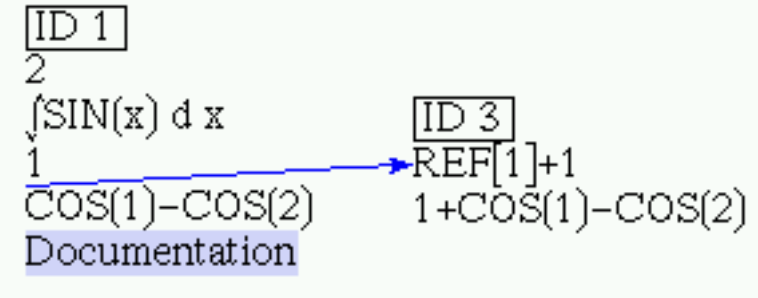

Figure 8: Example on referencing

There are some things to consider when referring to some‘thing’:

- ‘REF[1]’ is the correct and only way to use references.
- ‘REF[n]’, where n is not a number(but a variable, equation or whatever), will NOT work reliably.
- ‘REF[1]:=something’ is a bad idea. The results are unpredictable.
- ‘REF[1,n]’ doesn’t work (yet). Write REF[1][n] instead.

2.8 Editor commands

- Create a new ‘object’ with the middle mouse button (at the mouse position) or CTRL-N (below the last new expression).
- Create a new ‘object’ using the contents of the ‘Command’ entry field by pressing CTRL-r (at the current mouse position).

- Create a ‘link’ to the ‘Result’ subexpression selected by pressing SHIFT-CTRL-R.[8]

2.9 Flags

Objects, commands / modifiers and command / result lines can have associated properties. Most of them are accessed through the ‘properties’-dialog.

The ‘header’-line shows the ID and some flags affecting the whole object. Those are

- Result This shows, that the result is not up-to-date anymore. Press #Eval# to clear it. If it does not go away, there is an error in your calculation. **NOTE: This is obsolete. The box has been replaced by a red bar on the left of the object.**
- Result/Circular This result cannot be calculated, because there is a circular reference involved in its evaluation[9].
- Definition The command will be evaluated *before any other* evaluation[10]For example, turning a command %ENUMER:TRUE into a definition will allow the constant %E (2.71...) to be converted to a float number (By default, Maxima won’t give you a numerical result, if there is a %E in your expression).
- AutoExec The command will be evaluated each time you load the .mth-file. This is intended to regenerate plots as EPS-file.
- Recall only The command will not be evaluated. Useful to turn off objects, which produce errors.

[8]This only works on results! Also, if the structure of the source expression changes, the ‘link’ will probably not do what you expected. This command has an inherent ‘quick-and-dirty’-nature.

[9]Note: This does not necessarily mean, that this ‘thing’ is *within* a loop of circular references

[10]Of course, if you have several definitions, one of them has to be first. Do your calculation in a way, that the order of definitions does not matter.

2.10 Properties

Each ‘object’ has some associated properties.

- You can turn on or off any of the graphical elements. Use this to reduce screen clutter.
- Show also on interface page: If this is set, your expression appears also on the ‘interface page’ - see section 2.12.
- Function/variable definition: Here you can turn your expression into a ‘definition’ (see section 2.9).
- Eval repeatedly: When you hit the #Eval#-button, each command is run through Maxima wrapped in ev(YourCommand). Setting the EVAL flag turns this to ev(YourCommand,eval). See section 3.1.
- Eval numeric: Same as ev(YourCommand,numer). This will give a numerical result, if possible.
- Do NOT evaluate: This is the flag already explained in section 2.3. You can also use this to temporarily disable a command that produces an error or needs a lot of processing time.
- ‘Refs in eval’: Put here a list of IDs (separated by a colon), for example ‘42,24’. The referenced expressions will be included in the evaluation in the form ev(YourCommand,REF[42],REF[24]). Use this for substitution, maybe set the EVAL-switch.
- ‘Generate TEX-output, if possible:’ Setting this switch will create a bitmap representation for the output using TEX. You can scale it to (almost) any size. It is printed as postscript, not as bitmap.
- Picture: Path to an EPS-file. You can enter
  - ‘MyFile.eps’ - this file must be in the place where you store the .mth file or where you started Symaxx from (as obtained by ‘pwd’)
  - ‘MyDir1/MyDir2/MyFile.eps’ - relative path. ‘MyDir1’ must be in the place where you store the .mth file or where you started Symaxx from.

  - '/home/MyName/' - absolute path starting with /

- Scale: Changes size. The picture is printed as postscript, so the resolution on the printer is better than on the screen.

2.11 'Properties' dialog

Right-click any 'thing' and the 'Properties'-dialog pops up.

Right now, you have to press the ENTER-button anytime you have entered some new text anywhere, otherwise the change is not accepted.

2.12 Interface page

If you have a huge calculation and want to run it several times (changing just some parameters), you can use the 'Interface page' to get rid of screen clutter. Set the 'Also display on interface page'-flag for all the expressions that you want to change or read off. Press the 'Disp'-button. Everything else disappears (press 'Disp' again turns back to normal). This is the 'Interface page'. Note that you can move any'thing' to a new location, without changing the position on the original page.

2.13 Printing

You can print out your calculation into a postscript file (use 'ghostview' or send it to the printer by hand). The printed page shows exactly what you see on the screen. Graphics are embedded as EPS-files, so the printing quality will be as good as the original EPS file. The output file is always 'output.ps' in the same directory as the .mth-file (or, if you didn't save yet, the directory where you started Symaxx).

2.14 Units

Symaxx supports calculation with units as an additional feature. If the option 'Units' is not set in the 'Maxima' menu, there are no changes against standard Maxima.

Setting the Units-switch will

- Load the 'SymaxxUnits.mc' module. This is a modified 'units.mc'-file (located in Maxima's 'share' folder).
- highlight known units using blue colour
- replace the Maxima name of some units (NEWTON, JOULE, WATT etc.) with a typical abbreviation (N,J,W).
- Provide the functions ConvertUnit(Expr,Unit) and ConvertUnitNumer(Expr,Unit).

How to use units:

- Enter your numbers in whatever unit is convenient (10*FOOT+1*METER+3*INCH)[11]
- Upon evaluation all units are reduced to SI-units. If a conversion requires several steps, it is necessary to set the EVAL switch of the expression or use an extra ev(...,eval).
- Convert your results to whatever unit you require by using ConvertUnit(Expr,'Unit)[12]. The function will
  - Divide Expr by Unit
  - Try to simplify (cancel units)
  - ConvertUnitNumer will evaluate numerically at this point[13]
  - Multiply again with Unit

Typically there are several ways to write a physical unit (for example: 1 W=1 VA=Nm/s). The alternative would be to use only three base units (for example by using only meter, second, kilogramm, which is very unpractical when talking about volts and amperes).

Figure 9 shows an example for a calculation with units.

[11] Please note that all units names have to be typed in uppercase. Unit names are *no* built-in expressions. See section 3.8.

[12] Please remember to quote the unit, as in ConvertUnit(FOOT,'METER).

[13] Numerical evaluation was turned into an own function, because a target unit, which is composed of several 'base units' (for example Newton, Joule), will be reduced to base units again at the next evaluation.

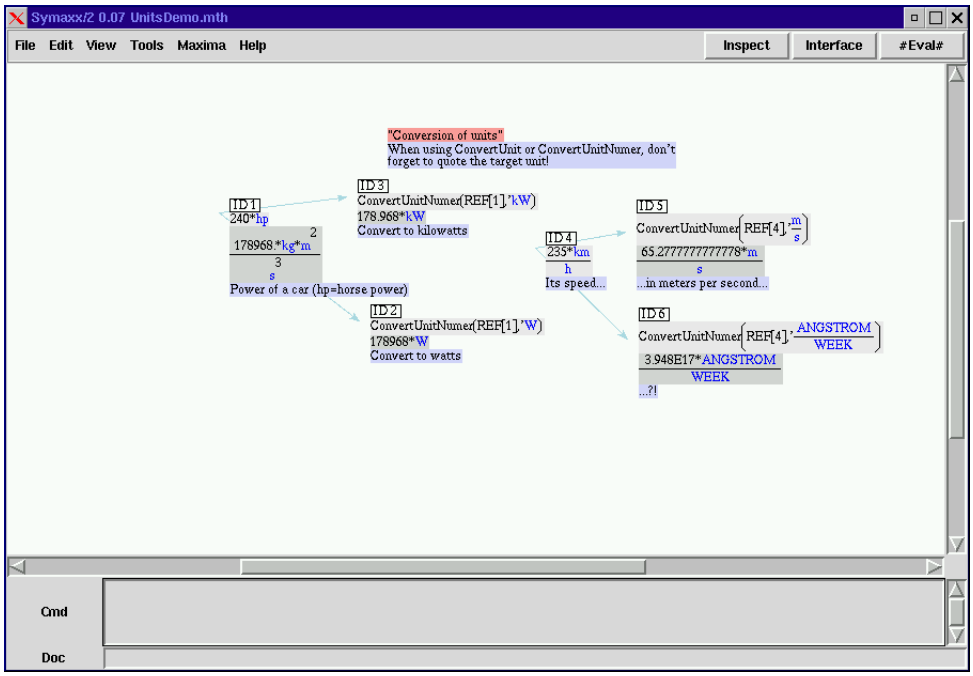

Figure 9: Calculation with units

3 Symaxx-Cookbook

Here I'll give some examples for common tasks.

3.1 Substituting values

Often, you have to replace one variable by something else:

- Another variable
- An expression
- A numerical value

For example, I'll replace $a,\ b,\ c$ and d in $\frac{a \cdot b}{c \cdot d}$ by $aa,\ bb,\ cc$ and dd, respectively. One way to do this in Maxima is to write

ev((a*b)/(c*d),a=aa,b=bb,c=cc,d=dd)

The implementation of this statement in Symaxx is shown in figure 10. I have put three x=y-lines into a list to demonstrate it with and without using a list.

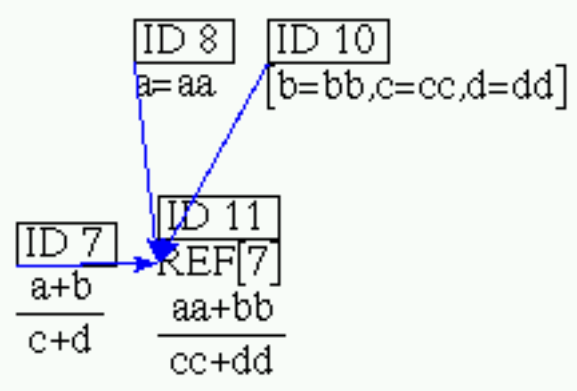

Figure 10: Replacing one variable by another

The references to ID 8 and ID 10 are done in the ‘Properties’ dialog (see also section 2.10), in the ‘Refs in eval’ field. Of course, you could also write

ev(REF[7],REF[8],REF[10])

as a command in ID 11.

Sometimes it is necessary to set the EVAL-flag. Maxima will then try the substitution lines for any number of times, until the result does not change anymore.

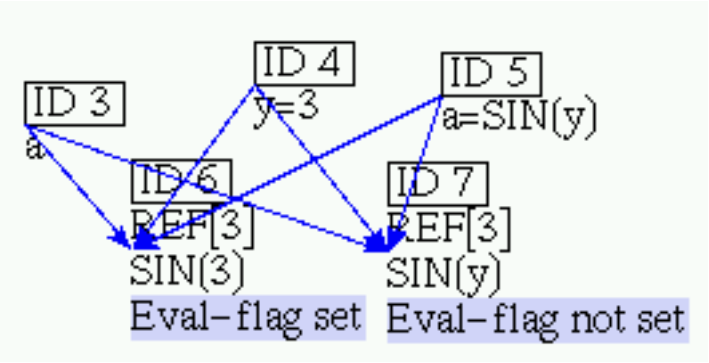

Figure 11: Example for EVAL-flag

Figure 11 shows this situation. The original expression is a, the substitutions are $y = 3$ and $a = \sin(y)$. The list of references is '4,5'.

- In ID 6, the first substitution does not match, only the second substitution turns a to $sin(y)$.
- In ID 7, the EVAL flag is set, the substitutions are repeated.

Not shown in the picture: Setting the NUMER flag in ID 6 results in 0.14112...

3.2 Pattern matching

Often it's useful to use pattern matching to transform expressions or pick out pieces (for example to find coefficients of the powers of a variable). Here is a very simple example:

BLOCK(

MATCHDECLARE(Amp,TRUE,Freq,TRUE)

,DEFMATCH(FreqAndAmp,Amp*SIN(Freq)))

This piece of code should be used as a definition or within a block. It creates a matching program 'FreqAndAmp(expr)'.

The 'MATCHDECLARE' statement tells Maxima that the variables 'Amp' and 'Freq' in the following DEFMATCH will match anything. TRUE could be replaced with a restriction.

The DEFMATCH statement defines a new rule called 'FreqAndAmp', which will match an expression $a * \sin(b)$ and return a list $[Amp = a,\ Freq = b]$.

Figure 12 shows the same. Figure 13 demonstrates, how to use it.

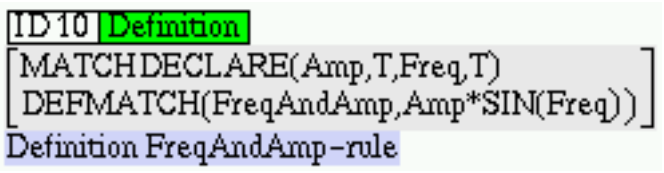

Figure 12: Pattern matching example: Defining the rule

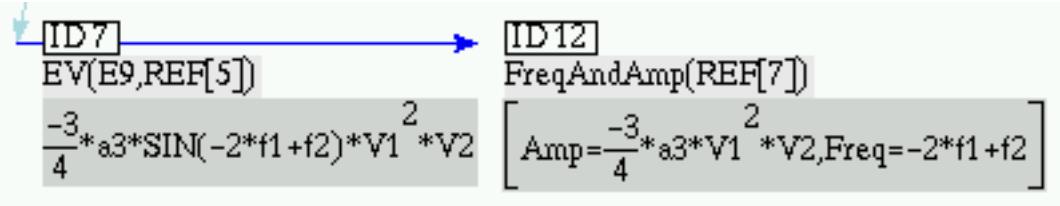

Figure 13: Pattern matching example: Using the rule

Here is another example. It creates a matching program ‘osz(expr,var)’.

BLOCK(

MATCHDECLARE(a,FREEOF(x),b,FREEOF(x),c,FREEOF(x)),

DEFMATCH(osz,a*SIN(b*x+c),x))

The FREEOF statement requires that the expressions a,b and c do not include the variable x.

The variable name ‘x’ is used only within the definition: Specify the variable you want to use for ‘x’ as a second argument to the matching program (figure 14).

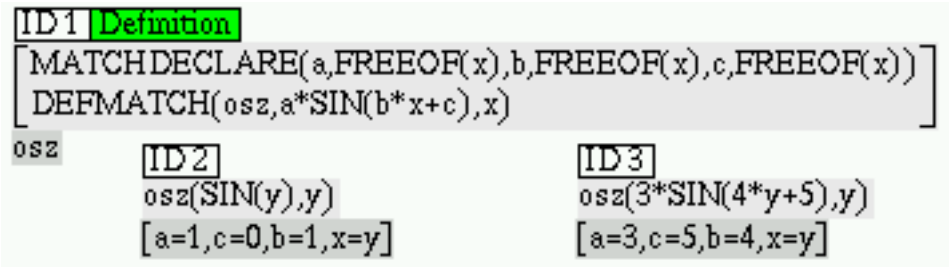

Figure 14: Pattern matching example 2

3.3 Using assumptions to prevent Maxima from asking questions

If Maxima needs to know something, it will ask. This is not supported in Symaxx, which will abort when it encounters any sort of question from Maxima. In this case, you have to give the answer in advance. Check the log to find out, what Maxima wants to know. Then create a new ‘assume’-statement and turn it into

a definition. Figure 15 gives an example for an integration, which requires an assumption.

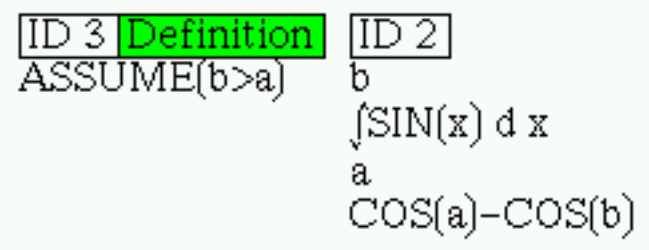

Figure 15: Example for ‘assume’-statement

If you don’t like the ‘global’ assumption, try the solution in figure 16.

```
BLOCK(ASSUME(c > d),INTEGRATE(SIN(x),x,c,d))
```

Figure 16: Another way to use the ´assume’-statement

3.4 ‘Pickapart’ complicated expressions

The ‘pickapart’ command will split up an expression into several expressions. Maxima creates additional labels internally, but Symaxx returns a list with two elements:

- The result of the ‘pickapart’-command
- A list of equations with the contents of all E-labels.

Figure 17 shows an example.

3.5 ‘Instant solve’ an expression

There is a built-in maxima program, which will try to solve an equation for an arbitrary subexpression. This function is invoked through the F10 key. It will create a new thing.

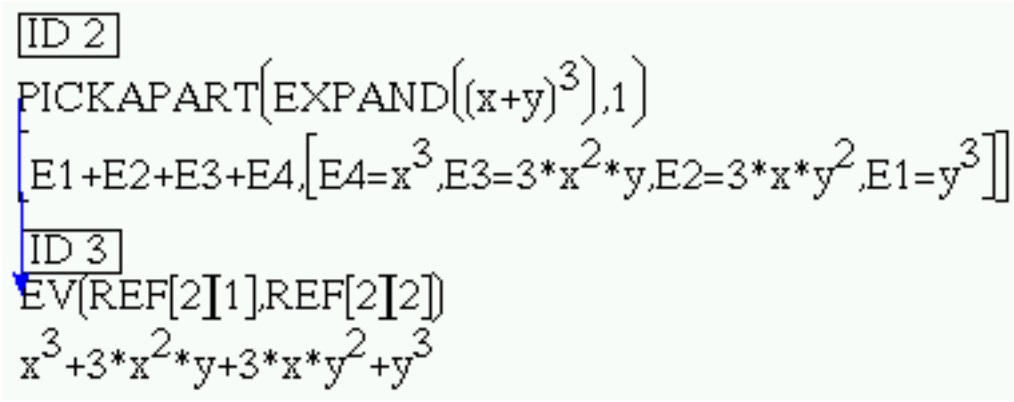

Figure 17: Pickapart and how to reassemble its output

It will substitute the subexpression you have highlighted with a temporary variable. It solves for that temporary variable and resubstitutes the original expression. Often it gives useful results, but sometimes the variable or expression you are interested in remain on both sides of the equation.

Figure 18 shows an example. The equation $\sin(\frac{x}{y}+z) = w$ is solved for z, $\frac{x}{y}+z$, $\frac{x}{y}$, x and y.

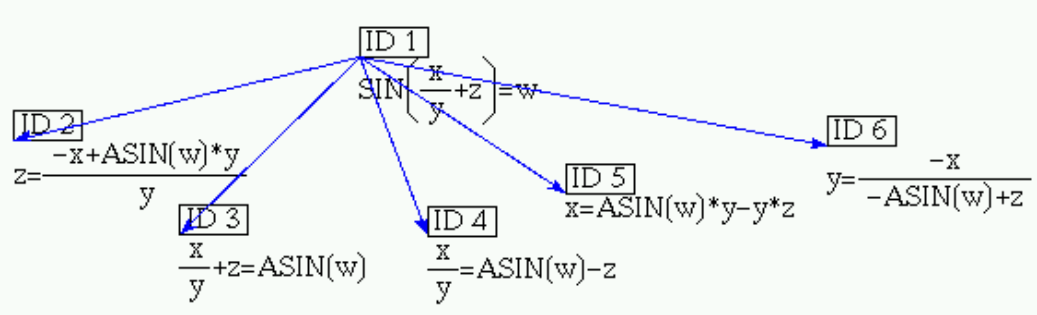

Figure 18: Instant solve example

3.6 Using piecewise-linear approximations

The predefined function LinInterp(x,PtList) gives a piecewise linear approximation through the points in PtList. A demonstration can be found in

/Demo/LinInterpDemo.mth.

To use this function, the ‘Feature’ *Use Numerical* has to be turned on in the main menu.

Figure 19 shows a plot for LinInterp(x,[[0,0],[1,1],[2,4],[3,9],[4,16]]).

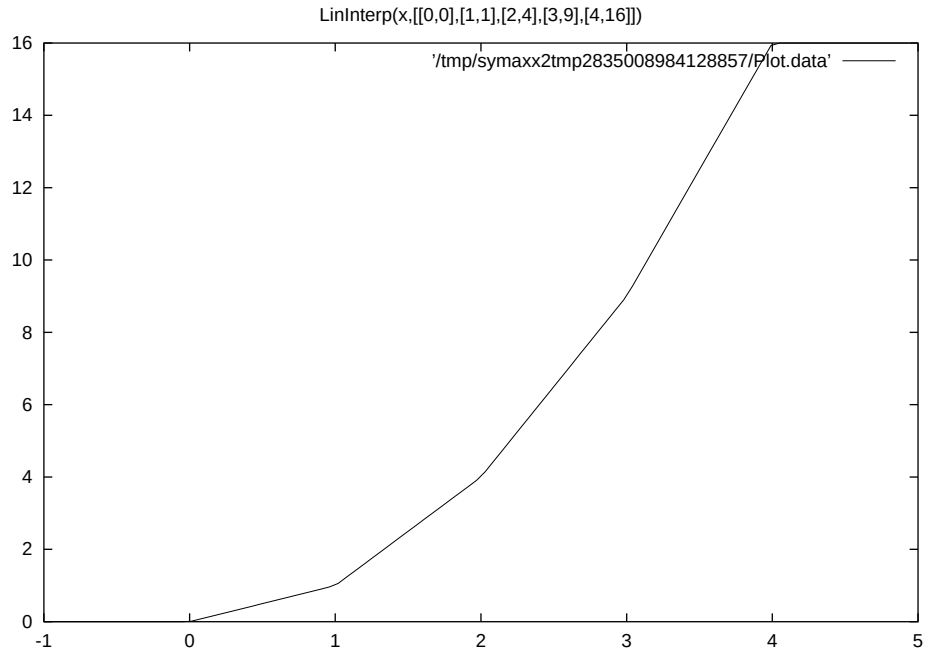

Figure 19: Piecewise linear approximation of x^2

3.7 Using numerical integration

Maxima provides the ROMBERG function for numerical integration. This can of course be used as it is. Alternatively, a wrapper function named ’NumIntegrate’ is available, which provides some error-checking. For this, the *Use Numerical* feature needs to be turned on.

Syntax:

NumIntegrate(Expr,Var,Start,Stop).

For example

NumIntegrate(Sin(x),x,0,6.28)

will return (almost) 0.

3.8 The meaning of uppercase and lowercase in Maxima[14]

- For *built-in expressions*, Maxima will automatically convert to uppercase. 'sin(x)','Sin(x)' and 'sIn(x)' are identical. Same for %PI and %pi, %i and %I etc.
- For *any other expression*, Maxima is case sensitive. MyFunc(x) and myfunc(x) are two completely separate functions.

3.9 Picking out subexpressions with INPART

When you hold down the left mouse button on any subexpression, the title line will show: INPART and then a series of numbers. The numbers are what you would have to give to the INPART function as arguments to pick out that subexpression[15].

There is a hidden trap, but you'll have to try hard to fall into it: By default, Maxima rearranges things (sums, products) using the default sorting rule (first z, then y, then x...). This mixes up subexpressions, the 'Path' to the wanted subexpression may no longer be correct. Sometimes you'll have to avoid this by temporarily turning off simplification, as in BLOCK([simp:off],INPART(MyFun(a+m+n+b),1,4)). This gives 'b' as a result. If your expression has been run through the simplifier before you got your INPART 'Path' (as is usually the case), you don't have to worry about this problem. This only occurs, when using INPART on an expression entered in the same line.

3.10 Using the YLIST function to enter lists

When Symaxx encounters an YLIST function, it will display the arguments stacked on top of each other. Figure shows, how it looks. YLIST(1,2,3,4) returns [1,2,3,4] after the evaluation.

[14] Symaxx does not alter Maxima's behaviour in any way regarding upper and lowercase.

[15] If you click on the toplevel expression, the line will only show INPART, which is logical: If you use the INPART function on an expression with no other argument, it will return the whole expression.

It is intended for entering lists (as an alternative way of writing [...]). See figure 20.

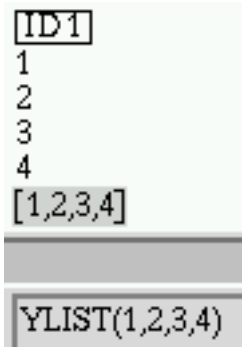

Figure 20: The YLIST-command

3.11 Debugging

There is no support for the Maxima-debugger. My recommendation is to open a normal Maxima session and debug functions there.

A major problem is that the output from the print-command is not visible anywhere. A possible way to obtain debug info is to use the LDISPLAY command, which will create an intermediate label for its argument. Those labels are then collected into a list (see section 3.4 for information on intermediate labels).

Here is an example (from the Maxima manual, section 'Program Flow'):

FOR A:-3 THRU 26 STEP 7 DO LDISPLAY(A)

It will create this output:

[DONE,[E6 = (A = 25),E5 = (A = 18),E4 = (A = 11),E3 = (A = 4),E2 = (A = -3)]]

4 Key shortcuts

- In any text entry field:
  - CTRL-C copies text to the buffer,

  - CTRL-X cuts it.
  - CTRL-V pastes the contents of the buffer.

- In the command entry field:

  - Shift-return processes the input line and evaluates
  - ALT-r inserts REF[],
  - ALT-p inserts %pi,
  - ALT-i inserts %i.

- Ctrl-return creates a line break in the documentation entry line.
- Ctrl-r creates a new ‘thing’ at the mouse location with the contents of the editor line.
- Ctrl-space evaluates.
- Some other shortcuts are shown in the menus.

5 ‘Features’

Symaxx contains a couple of add-ons, that provide new commands and functionality. Any ‘feature’ has to be turned on before it can be used, to avoid wasting too much computation power on loading unneeded modules.

5.1 Units

This includes a slightly modified /share/units.mc file. Many common units are recognized and drawn in blue. For example, typing CENTIMETER will show a blue ‘cm’.

5.2 Selection

The idea of ‘selecting’ subexpressions is to protect them against any further manipulation. There is a custom-built ‘simplifier’, that will merge the sum or product of two ‘selected’ expressions, if they are selected into the same ‘group’ and the ’SELSIMP’ flag is set.

(doc not yet written, turn on and right-click a command line)

Note: The implementation is not fully complete. It’s useful sometimes, but there are some open issues.

5.3 Gnuplot

Symaxx provides a simple interface to create two-dimensional ‘gnuplots’. This is done in two steps.

- First a ‘curve’ is generated from a given function, a variable and range and nr. of points, a title, and a connection style.
  This creates a data object, which is shown as #PLOT#, or on closer examination as SYMAXX_GNUPLOT_DATA(...).
  This is done in the ‘Properties’ dialog, section ‘Line’ - ‘Gnuplot-curve’.
  The function can be a list with two elements, which results in a parametric plot (for example [sin(x),cos(x)] results in a circle).
- Then one or more curves (combine several #PLOT# objects into a list) are plotted into an EPS file. Some options (axes labels, scale, grid,...) can be set here.
  Look under ‘Properties’, section ‘Line’-‘Gnuplot-diagram’.
- Both steps can be done in one line.
- Example:
  - Enter a function x^2
  - In ‘Properties’, ’Line’, ‘Gnuplot-curve’, set the variable to ‘x’, ‘Min’ to ‘-1’, ‘Max’ to ‘1’, ‘Nr. Points’ to ‘101’.
    Enable ‘Create Gnuplot data’.
  - In ‘Properties’, ’Line’, ‘Gnuplot-diagram’, enable ‘Create plot from data’.

5.4 Format

This includes the ‘format’ package, which does not come with Maxima. For example, FORMAT(expr,%POLY(x)) will rearrange expr as a polynomial in x. *Very* useful.

Original file from http://math.nist.gov/~BMiller/computer-algebra/ (documentation there).

Important: It is necessary to run

symaxx -compile

from the command line before using it - this will compile the needed LISP files. This can even be done, while a Symaxx-session is already running. You need write access to the Symaxx installation folder.

5.5 Numeric

Provides the functions LinInterpolate, NumIntegrate, log10 (logarithm to the base of 10).

6 Miscellaneous

6.1 Odd behaviour of selection in editor

In some situations, you cannot select the numerator of a fraction. This happens, when Maxima returns x^{-n}, which is drawn as $\frac{1}{x^n}$. Simply edit the whole expression ‘1/x^n’.

In a derivation, the order n appears in both numerator and denominator: $\frac{\mathrm{d}^n f(x)}{\mathrm{d}x^n}$. When you click on the n in the denominator, the n in the numerator gets selected instead. This is not a bug, it’s a feature... (Both must be the same, anyway).

6.2 Error in evaluation

If Maxima aborts with an error, the ‘Log’ button will light up. You can have a look at the transcript. It helps to find out the reason for the error (hint: start looking at the end). Sometimes the button does not light up, even though an error occured.

If you enter a syntactically invalid expression like '1+'[16], Symaxx will refuse to store it. Either correct it or discard it by selecting anything.

6.3 Killing Maxima / console interrupts

Sometimes Maxima gets stuck. Usually a calculation takes too much time. In this case Symaxx hangs.

After one second, a 'Stop'-button appears. Click it and Maxima will be killed.

If this doesn't work for any reason, kill the 'saved_maxima'-process with 'kill -9'[17]. Symaxx will then recover. You can also press CTRL-C on the console, where you have started Symaxx. This will kill Maxima, if Maxima is running. Otherwise, it does the same as the 'quit'-function (it will ask, if you have not saved).

6.4 Fixing the 'tex' function

(Not implemented anymore - obsolete)

Symaxx requires the Maxima 'tex()' command to produce high-quality output. The lisp-file has to be installed manually, although it comes with the Maxima source:

cd /usr/src/packages/SOURCES/Maxima/maxima-5.x (or wherever your Maxima source is located)

cp src/mactex.lisp (wherever your Maxima is installed)/maxima-5.x/src/

Otherwise, setting the 'Generate TEX-output' flag will have no effect.

You can check the 'tex' function by running 'maxima' by hand and typing 'tex(3*4)'. If you don't get an error message, it works.

6.5 Avoiding trouble

There are some things to remember:

[16] Non-matching brackets is a typical example

[17] Depending on your operating system, a 'killall saved_maxima' will achieve this.

- You can enter commands in a way the Maxima parser would not allow (for example by modifying the NIL in figure 3). Maxima is not foolproof. Neither is Symaxx.

- Symaxx cannot handle 'Rational' expressions in CRE form (see Maxima manual, section 'Polynomials'. They are converted to normal form automatically[18]. For example, RAT(3) will result in the same '3' that you get when you just type '3'. (Maxima-internally, they are not the same).

- In my experience, Symaxx is already quite safe. But save often and under different file names.

- System security: When a .mth-file Symaxx is loaded, it is executed as perl-code[19]. By manipulating a .mth-file, any command could be executed at this time. The risk is about the same as if you compile other people's code (for example, a ./configure-script has the same 'power'.)

6.6 Preferences

The file 'Symaxx2/Preferences.pm' contains many settings (for example default window and font size, where to find documentation,...). The administrator can edit them.

- Executable paths: Here you can set a command or a full path for Netscape, Acroread and Maxima. A list of programs is given for the .pdf viewer and the .html browser. The first entry that can be found will be executed when you open the manual or help. If you use the Maxima documentation a lot, I recommend to change it to 'lynx' (fast and small).

- Do not round: By default, Symaxx will chop off a single trailing digit, if there are at least 6 '0's in front[20]. This can be turned off with this switch.

- Fontsize: Set whatever is convenient for you. Some font sizes work better than others.

[18] Conversion is done with the 'totaldisrep' function on any data handed back to Symaxx.

[19] I could fix this if I'd write an own parser for '.mth' files. The present solution will probably be faster, though.

[20] This does not affect the accuracy of the calculation itself, only the display data is rounded.

- Paper geometry for postscript output: If you are using non-A4 paper, change these settings.
- Watchdog enabled: If you have a small machine (slow and not enough memory), you can turn off the 'Watchdog' to preserve resources.

6.7 Installation instructions

Starting with the archive Symaxx_something_tar.gz, do a

gzip -dc Symaxx_something_tar.gz | tar -cvf -

cd Symaxx2

symaxx

This is all as long as the necessary programs are installed (section 8).

If perl is not in /usr/bin, you'll have to change the first line of the files 'symaxx' and 'Symaxx2/Watchdog' to the location on your system, as found by 'whereis perl'.

6.8 Installing Tk

Do a

gzip -dc Tk800.022.tar.gz | tar -xvf -

cd Tk800.022

perl Makefile.PL

make

make test (optional)

make install

if you don't have root access, use

perl Makefile.PL LIB=/home/(place to install) or

perl Makefile.PL PREFIX=/home/(place to install).

make; make install

Find the location of the folder ‘Tk’ in the folder you gave as argument to LIB or PREFIX. Now you’ll have to change the first line in ‘symaxx’ and ‘Symaxx2/Watchdog’ to include the Tk library. Append a

-I /home/(where you installed Tk)/ (The folder that contains Tk)

as in

#!/usr/bin/perl -w -I /home/somewhere/

7 Conventions / differences to standard Maxima

- no labels C\d+. D\d+
- E-labels E\d+ are handled automatically: see section 3.4
- *SymaxxWorkDir* contains working directory
- *SymaxxFileName* contains name of current file without .mth-extension. sconcat(SymaxxWorkDir,SymaxxFileName,”.mth”) will give the full path of the Symaxx file.
- *SymaxxTmpDir* contains the name of the temporary directory. The contents will be deleted, when Symaxx ends.
- *REFASCII\d+* : reserved word (\d+: one or more digits)
- *REFINRES\d+*: reserved word
- REF[\d+]: Reference to ID \d+
- totaldisrep(expr) is done after each evaluation. Symaxx can’t handle expressions in (internal) CRE form.
- The function *gnuplot*(fun,var,[min,max],pts,”file.eps”) will generate a postscript plot in the current working directory.
- The function LinInterp(var,PtList) will calculate a piecewise linear approximation of PtList - see section 3.6.
- The function InstantSolve(Expr,Subexpr) will try to solve Expr for Subexpr.

- The function log10(x) is defined as log(x)/log(10). It will return $\log_{10}(x)$.
- YLIST(): See section 3.10.
- ErrorReporter(): reserved function
- $f^n(x)$ is drawn as $f(x)^n$. For example, $\sin^2(x)$ is drawn as $\sin(x)^2$. Please don't mix this up with $sin(x^2)$!
- A global definition 'simp:off' will not work. Use the flag in the 'properties' dialog or write BLOCK([simp:off],YourExpression) instead.
- Reserved names: All words, that start with SYMAXX.

8 System requirements

You'll need

- Unix - I use Linux
- perl - I am using 'perl, version 5.005_03 built for i586-linux', check with 'perl -v'. Download from 'cpan.org', should come with your OS distribution.
- maxima - check with 'maxima' (I use 5.5 beta, sometimes 5.4). Download from http://www.ma.utexas.edu/users/wfs/maxima.html
- perl/Tk - I am using version '800.014' and '800.022'. Check with 'widget' - Help - About. Download from 'http://www.cpan.org/authors/id/N/NI/NI-S/Tk800.022.tar.gz' (Or install from your Linux distribution).
- netpbm - test with 'pstopnm' - download from 'http://sourceforge.net/projects/netpbm' (Or install from your Linux distribution)[21].
- TEX and xdvi and dvips (comes with a LATEXdistribution)[22]
- A lot of memory (64 MB are recommended and sufficient). Don't even think about running Symaxx on your PDA.

[21] Symaxx runs without this, but no EPS graphic inclusion and no TEX output
[22] Symaxx runs without TEX, but there is no 'pretty' TEX output

9 Contact

nentwig@users.sourceforge.net