IMEKO 22nd TC3, 12th TC5 and 3rd TC22 International Conferences
3 to 5 February, 2014, Cape Town, Republic of South Africa

UNCERTAINTY CALCULATIONS USING FREE CAS SOFTWARE MAXIMA

Dirk Röske

Physikalisch-Technische Bundesanstalt (PTB), Braunschweig, Germany, dirk.roeske@ptb.de

Abstract: There are many computer software packages available for the calculation of measurement uncertainties. Some are free and open source, but many are not free to use or are restricted to non-commercial use. Not in all cases, however, is it necessary or advisable to purchase and learn to use a more or less fully featured software package. Some simple calculations can be done using free spreadsheet software contained in LibreOffice or OpenOffice. Also, MS Excel is often already available on the PC.

In some other cases the calculations are not simple, especially when the partial derivatives of the model functions have to be calculated. It would then be a good idea to use a free computer algebra system (CAS). This software facilitates symbolic mathematics.

This paper shows how to use the free CAS Maxima for calculating uncertainties of the torque values realized in a torque standard machine. The tool can also be used for the independent verification of uncertainty calculation results. Some additional tools can make the use of Maxima even simpler.

Keywords: uncertainty calculation, computer algebra system, CAS, Maxima.

1. INTRODUCTION

Wikipedia offers a table with 26 software tools for the calculation of uncertainties [1]. Half of these tools are either free (4) or available under a GNU GPL (6), BSD (2) or CC (1) license, the others are proprietary software or tools with a restricted or an unknown license. Some tools are available for different platforms; some have a special application, for example, the analysis of quantitative real-time polymerase chain reactions. The programming languages range from C/C++ via Delphi, Perl and Python to JavaScript.

For a user who needs to calculate the uncertainty of an output quantity in the case that the model and the values of all input quantities including their uncertainties are known, it is not easy to find the right tool. This becomes even more obvious when the user is not very familiar with uncertainty calculations and programming languages.

In this paper we will try to show how the free mathematical software package Maxima [2] can be adapted for the task of calculating measurement uncertainties by the example of torque values realized in a torque standard machine.

2. THE ADVANTAGE OF CAS

Calibration guidelines often contain quite detailed information on how to calculate the measurement uncertainty of a calibration result. Let us assume the use of the EURAMET cg-14 “Guidelines on the Calibration of Static Torque Measuring Devices” [3]. The model used in this guide is given in (1):

$$\bar{X} = (S + \delta S_{b'} + \delta S_b) \cdot M_\mathrm{k} + \delta X_r \qquad (1)$$

Here, the output value $\bar{X}$ depends mainly on the sensitivity S – the change of the output signal X in mV / V when a calibration torque M_k is applied. During the calibration, additional parameters of a torque transducer, like repeatability b', reproducibility b, and resolution r are defined. From these parameters, the standard uncertainties can be calculated in N·m taking into account the evaluation type and distribution function (see Table 1).

Quantity	Evaluation of standard uncertainty	Standard uncertainty in N·m
Repeatability in unchanged mounting position b'	type A	$u_{b'} = \frac{b'}{S \cdot \sqrt{2}}$
Reproducibility in different mounting positions b	type A	$u_b = \frac{b}{S \cdot \sqrt{2}}$
Resolution r	type B with rectangular distribution	$u_r = \frac{r}{\sqrt{12}}$
Reference torque	type B	u_tcm

Table 1: Input quantities, their type of uncertainty evaluation and standard uncertainties.

When a straight line fit is applied to the results, the standard uncertainty $u(\bar{X})$ of the random variables expressed in units of indication is calculated for each calibration step according to (2):

$$u(\bar{X}) = S\sqrt{u_\mathrm{tcm}^2 + u_{b'}^2 + u_b^2 + 2 \cdot u_r^2} \qquad (2)$$

It can be seen that only basic mathematical functions (sum, division, squares, and square roots) are required to calculate these values.

Let us now assume the use of a model for the calculation of the torque M realized in torque standard machines using dead weights and a supported lever (3):

$$M = m \cdot g \cdot l \left(1 - \frac{\rho_{\text{air}}}{\rho_m}\right) \cdot \cos(\alpha) + \sum_{i=1}^{n} \Delta M_i \qquad (3)$$

with the uncorrelated quantities

- m - the total mass of the active weights,
- g - the local gravitational acceleration on site,
- ρ_{air} - the density of the surrounding air,
- ρ_m - the density of the material of the weights,
- α - the lever's inclination angle against a horizontal line,
- ΔM_i - up to n additional influencing quantities.

For the calculation of uncertainties of a torque standard machine, the author considered three additional quantities: frictional torque in the air bearing ΔM_R, altering driving torque in the bearing ΔM_A, and electrostatic forces acting on weights or the lever ΔM_F.

In the case of model (3), the uncertainty of the calculated torque M must be calculated according to GUM [4], i.e. the partial derivatives of the model function must be calculated. This can be done using a computer algebra system (CAS) like Maxima.

3. THE USE OF MAXIMA

Maxima can be used as a pure command line tool by entering commands at the program's prompt. The better way, however, is to use the graphical user interface (GUI) wxMaxima. It offers a menu with a variety of commands and dialogs as well as clickable symbols. It can be effectively used to explore different commands and see the results produced by Maxima. But in both cases the use of Maxima is not very user-friendly and productive if many values need to be calculated. The aim of this study is to collate all values and graphs using a programmed file containing all necessary steps.

The first attempt might be a Maxima session that is stored in a file. The program offers various file types and extensions for this purpose. The "Maxima batch file" with the extension "mac" is best suited for our objective. Such a file can be edited using, for example, the free Notepad++ [5].

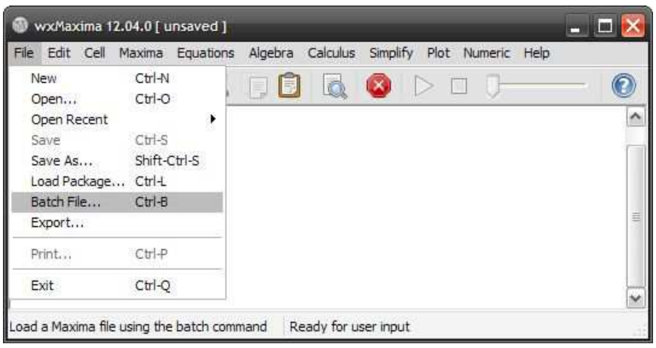

Figure 1: wxMaxima program window.

Each command in the .mac file must be followed by one of the two characters: semicolon (;) or $. The first means that Maxima's answer is displayed as output; the latter is used for a silent mode when no output is generated.

A .mac file can be opened by double-clicking on it provided that the operating system knows which program has to be used with the given file extension. If the program has been set up properly, the file will normally be opened by wxMaxima with the "load" command. The file is executed then without being displayed. The user must take care that results are written to files, or outputs like graphs are generated. Another option is to open the .mac file from the program's file menu using the command "Batch file…". The file can be selected using the corresponding OS dialog. Closing the dialog with the "Open" button lets Maxima open and execute the file. All input commands and results (depending on the character following the command, $ or ;) are displayed.

The very minimum one needs to know about Maxima batch files in the given context is summarized in Table 2.

Operation | Notation **Example**
Assignment of a value to a variable (g0) | : g0: 9.812524
Assignment of a list of values to a variable (m0List) | : […, …] m0List : [0.00040773594, 0.004077139, 0.040770601]
Definition of a function | define(f(x1, …, xn), expression) define(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), m*g*l*(1-rhoA/rhoM)*cos(alpha) + mR + mA + mF)
Definition of a list | : makelist(expression, n, n0, nmax) MD0List : makelist(), generates an empty list
Appending a value to a list | append(listname, expression) MD0List : append(MD0List, [MD0])
Plotting a 2d graph | plot2d (…) plot2d ([discrete, MD0List, yList],[style, points])

Table 2: Some basic operations in Maxima.

The definition of the function MD in Table 2 (4^{th} row) is a direct transcription of (3) into the programming language. If more than one result has to be calculated, then the program flow functions of Maxima can do this. The full script for calculating many (in this example: four) results is given in Fig. 1. It consists of less than 60 lines of code including empty lines but also lines containing more than one command (lines 1, 34, and 57). The program flow is realized in line 36 using a 'for' command, doing an iteration over the index variable i from $i = 0$ to $i = 3$. The calculation instructions for each value of i are enclosed in parentheses '(' and ')' and single commands are separated by a comma ','.

The capabilities of a CAS are used in lines 24 through 32 where the partial derivatives of the function MD are defined symbolically. This is the difference to some other tools like, for example, Excel, where the derivatives must be known as a function during programming in order to get their values calculated at the given points. Maxima does the calculation at runtime and there is no necessity to do this calculation by hand. Furthermore, if the starting function (MD) has to be changed, only line 23 is affected. Again, there is no necessity to change the derivatives of this function except in the case that the number or names of the independent variables of this function change.

The uncertainty values W are calculated in line 52 and stored in the list 'yList' (line 54) for further use in the plot command (line 59). W is calculated according to

$$W(x_1^0, x_2^0, \dots x_n^0) = \frac{2}{f} \cdot \sqrt{\sum_{i=1}^{n} \left(\frac{\partial f}{\partial x_i}\right)^2 u^2(x_i)} \qquad (4)$$

with the input quantity estimates x_i^0 and the model function (3)$f(x_1, x_2, \dots x_n) = M(m, g, l, \rho_{\text{air}}, \rho_m, \alpha, M_R, M_A, M_F)$. The output graph and the results are shown below in Figs. 4 and 5, but first possible improvements will be considered.

4. ADDITIONAL TOOLS

Figure 1 shows the well-structured source code with aligned blocks so that typing errors can be found easily. As can be seen, the code contains large blocks with the same data (coloured frames in Fig. 2) and the question is whether this can be used for a simpler code generation.

For this purpose, another free tool is used: AutoIt [6]. AutoIt is a programming environment for writing and executing Basic-like scripts that can be used to automize any work with the Windows operating system. For example, it can be used for an automated generation of the Maxima input file without the necessity of programming.

In this example – the calculation of torques generated by a torque standard machine and the determination of the corresponding relative expanded uncertainties – nine input

```
1  kill(all) $ load("basic") $
2  
3  m0List : [0.00040773594,0.004077139,0.040770601,0.4077028]$
4  g0     : 9.812524$
5  rhoA0  : 1.2$
6  rhoM0  : 7975$
7  l0     : 0.25$
8  alpha0 : 0$
9  mR0    : 0$
10 mA0    : 0$
11 mF0    : 0$
12 
13 umList : [0.00000001,0.00000002,0.00000003,0.00000027];
14 ug     : 0.000005$
15 urhoA  : 0.0462$
16 urhoM  : 6.09$
17 ul     : 0.000005$
18 ualpha : 0.000714$
19 umR    : 0.00000029$
20 umA    : 0.00000029$
21 umF    : 0.00000029$
22 
23 define(MD       (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), m*g*l*(1-rhoA/rhoM)*cos(alpha) + mR + mA + mF)$
24 define(dMD_m    (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), m))$
25 define(dMD_g    (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), g))$
26 define(dMD_l    (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), l))$
27 define(dMD_rhoA (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), rhoA))$
28 define(dMD_rhoM (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), rhoM))$
29 define(dMD_alpha(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), alpha))$
30 define(dMD_mR   (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), mR))$
31 define(dMD_mA   (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), mA))$
32 define(dMD_mF   (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), mF))$
33 
34 MD0List : makelist() $ yList : makelist()$
35 
36 for i: 0 thru 3 do (
37 m0 : pop(m0List),
38 um : pop(umList),
39 
40 MD0         : MD       (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0),
41 
42 cMD_m       : dMD_m    (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0)    *um,
43 cMD_g       : dMD_g    (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0)    *ug,
44 cMD_l       : dMD_l    (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0)    *ul,
45 cMD_rhoA    : dMD_rhoA (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0) *urhoA,
46 cMD_rhoM    : dMD_rhoM (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0) *urhoM,
47 cMD_alpha   : dMD_alpha(m0, g0, l0, rhoA0, rhoM0, ualpha, mR0, mA0, mF0)*ualpha,
48 cMD_mR      : dMD_mR   (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0)   *umR,
49 cMD_mA      : dMD_mA   (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0)   *umA,
50 cMD_mF      : dMD_mF   (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0)   *umF,
51 
52 W           : 2*sqrt(cMD_m^2+cMD_g^2+cMD_l^2+cMD_rhoA^2+cMD_rhoM^2+cMD_alpha^2+cMD_mR^2+cMD_mA^2+cMD_mF^2)/MD0,
53 MD0List     : append(MD0List, [MD0]),
54 yList       : append(yList, [W])
55 );
56 
57 MD0List; yList;
58 load(draw)$
59 plot2d ([discrete, MD0List, yList],[style, points],[xlabel, "Torque in N·m"],[ylabel, "Rel. exp. unc."],[logx],[logy])$
```

Figure 2: .mac file for input in wxMaxima, screenshot taken from Notepad++.

```
20  umA      : 0.00000029$
21  umF      : 0.00000029$
22
23  define(MD         (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), m*g*l*(1-rhoA/rhoM)*cos(alpha) + mR + mA + mF)$
24  define(dMD_m      (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), m))$
25  define(dMD_g      (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), g))$
26  define(dMD_l      (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), l))$
27  define(dMD_rhoA   (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), rhoA))$
28  define(dMD_rhoM   (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), rhoM))$
29  define(dMD_alpha  (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), alpha))$
30  define(dMD_mR     (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), mR))$
31  define(dMD_mA     (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), mA))$
32  define(dMD_mF     (m, g, l, rhoA, rhoM, alpha, mR, mA, mF), diff(MD(m, g, l, rhoA, rhoM, alpha, mR, mA, mF), mF))$
33
34  MD0List : makelist() $ yList : makelist()$
35
36  for i: 0 thru 3 do (
37  m0 : pop(m0List),
38  um : pop(umList),
39
40  MD0         : MD        (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0),
41
42  cMD_m       : dMD_m     (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0)     *um,
43  cMD_g       : dMD_g     (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0)     *ug,
44  cMD_l       : dMD_l     (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0)     *ul,
45  cMD_rhoA    : dMD_rhoA  (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0) *urhoA,
46  cMD_rhoM    : dMD_rhoM  (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0) *urhoM,
47  cMD_alpha   : dMD_alpha (m0, g0, l0, rhoA0, rhoM0, ualpha, mR0, mA0, mF0)*ualpha,
48  cMD_mR      : dMD_mR    (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0)   *umR,
49  cMD_mA      : dMD_mA    (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0)   *umA,
50  cMD_mF      : dMD_mF    (m0, g0, l0, rhoA0, rhoM0, alpha0, mR0, mA0, mF0)   *umF,
51
52  W           : 2*sqrt(cMD_m^2+cMD_g^2+cMD_l^2+cMD_rhoA^2+cMD_rhoM^2+cMD_alpha^2+cMD_mR^2+cMD_mA^2+cMD_mF^2)/MD0,
53  MD0List     : append(MD0List, [MD0]),
54  yList       : append(yList, [W]),
```

Figure 3: .mac file with marked command blocks containing the same information in each line.

quantities are considered. In most of the calculations the number of these quantities is not very large so that a form can be used for the input. AutoIt has many tools for different programming requirements; among them is an editor for graphical user interfaces (GUIs) which can be used for the generation of such an input form for data used for the calculations.

The structure of the script in Fig. 1 is very simple:

Line	Explanation
1:	Maxima internal commands
3 – 21:	input quantities and their standard uncertainties are defined (single value or list of values)
23 – 32:	model function and its partial derivatives defined
34:	list of results defined
36 – 55:	iteration over list of input values
37 – 38:	value fetch for the iteration
42 – 50:	calculation of intermediate values
40, 52:	calculation of the result(s)
53 – 54:	storing of the result in a list for further use
57:	display of the results in wxMaxima
58:	Maxima internal command
59:	graphical output as 2D plot

In fact, the only inputs that have to be carried out by the user are the model function and the input quantities including their names, values and standard uncertainties. All the other steps that are necessary to calculate the results follow from the GUM and are in general independent of the model and the input quantities. That is why it is possible to write a program that can be applied to any model function with a freely definable number of uncorrelated input quantities. For practical reasons, a maximum number of 25 should be sufficient.

NOTE: A careful look inside the blue frame in Fig. 3 reveals a deviation in the sixth column: among all the ‘alpha0’ there is one ‘ualpha’. The reason for that is the value of the derivative of the cosine function: for an angle of zero this value is also zero, so the inclination angle does not contribute to the uncertainty budget. A simple way to include this contribution is to calculate the value of the partial derivative not at $\alpha = 0°$ but at the angle given by the standard uncertainty. For more details see [7]. This special case was not implemented here.

A program and GUI were composed by the author. They allow the model function to be written in the format that can be interpreted by Maxima on the first tab (see Fig. 4).

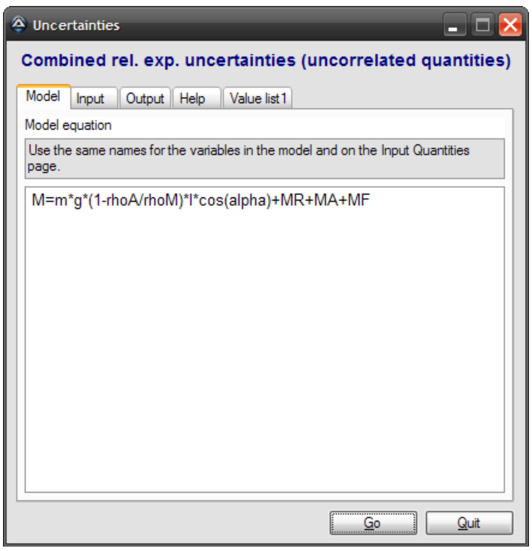

Figure 4: Definition of the model function.

All the input data can be determined with their name, value, and corresponding standard uncertainty on the Input tab (Fig. 5). If this data consists of more than one value – defined by the checkbox under 'List' – the list of values and uncertainties can be inserted on a special page (Fig. 6).

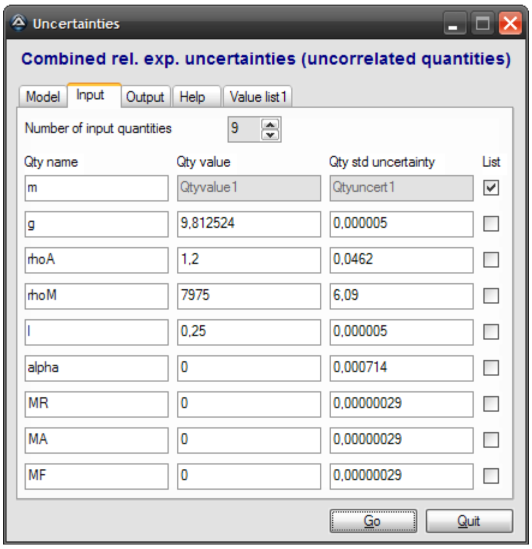

Figure 5: Definition of the input quantities: their number can be adapted; the names should be the same as those used in the model function.

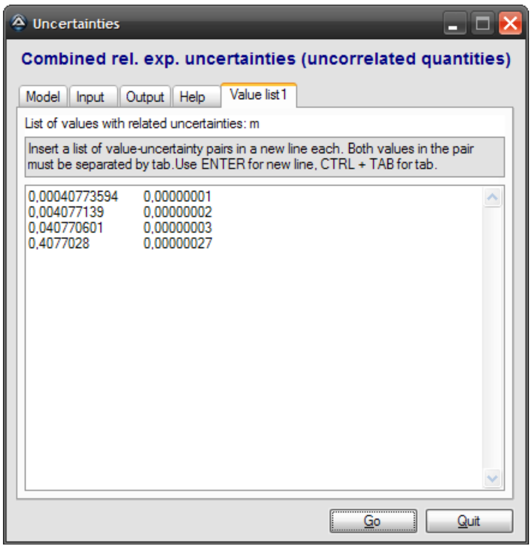

Figure 6: Input quantities with many values on a special page.

In order to use this tool with different models and values of the input quantities, the file name of the output file can be changed. Some other parameters are adjustable, for example, the axis labels of the plot that is generated by Maxima (Fig. 7). Some other details will be implemented too, for example, whether the scaling of the axes should be logarithmic or not. Because of the large variety of possible parameters, their number should be limited to the most important ones for this tool.

For the sake of completeness and for better usability, the program has a Help tab (Fig. 8) with some very basic explanations on how to use it.

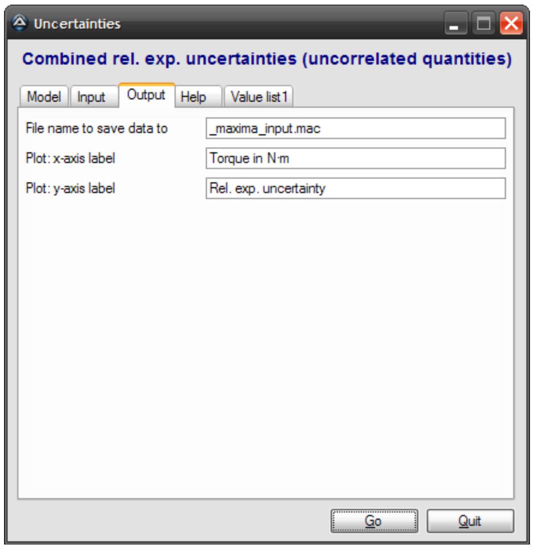

Figure 7: Settings for output.

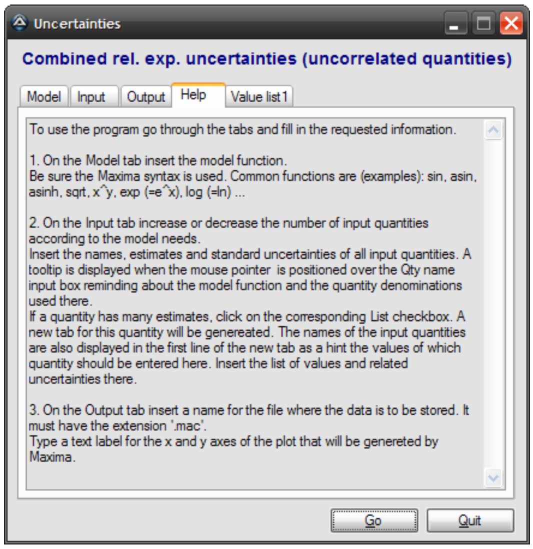

Figure 8: Help tab of the program.

A problem may arise if the user does not remember the correct name (or its spelling) of a variable used in the model on the Model tab when wanting to enter the data on the Input tab. The tool is not able to recognize variables. As a hint to the user, a tooltip with the model function is displayed when the mouse pointer is positioned over one of the Qty name input boxes.

As can be seen in Fig. 6, the name of the tab for a quantity with a list of values is 'Value list 1'. Number 1 is the ordinal number of the value on the Input tab. If there are a larger number of such quantities, the name of the tab may be 'Value list 8' and it is not obvious to which quantity it belongs. For this purpose, the name of the quantity is given at the top of the tab: in the case of Fig. 6 it is 'm'.

5. RESULTS

Figure 9 shows the plot generated by Maxima. It may be formatted according to the needs of the user using the available Maxima commands. For this tool, a simple *x-y* plot

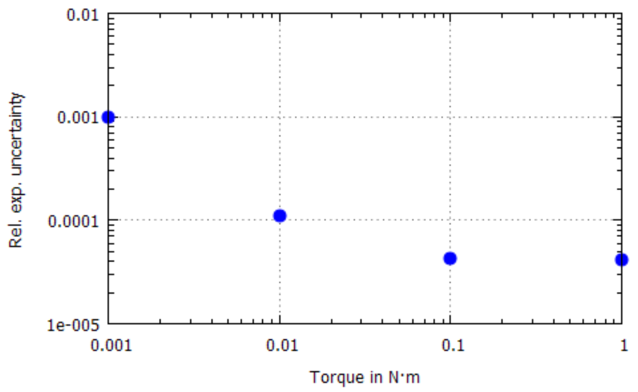

Figure 9: Plot by Maxima: rel. exp. (k = 2) uncertainties of the torques generated in a torque standard machine in the range from 1 mN·m to 1 N·m.

with blue dots was chosen. Maxima offers many more plotting functions and parameters using mainly the gnuplot [8] library which is usually installed together with Maxima.

If it is necessary to obtain the numerical values, then the program window of Maxima may be consulted, see Fig. 10. For this purpose the input file must be opened using the "Batch file …' command (see Fig. 1). The torque and the corresponding uncertainty values are contained in the lists MD0List and yList. It is possible to copy these values by hand.

The program presented above stores all values in a file using the 'stringout' command, but Maxima also offers other ways to save data to files. The value of a variable called 'file_output_append' determines if the data is appended to the file or if the existing file will be truncated.

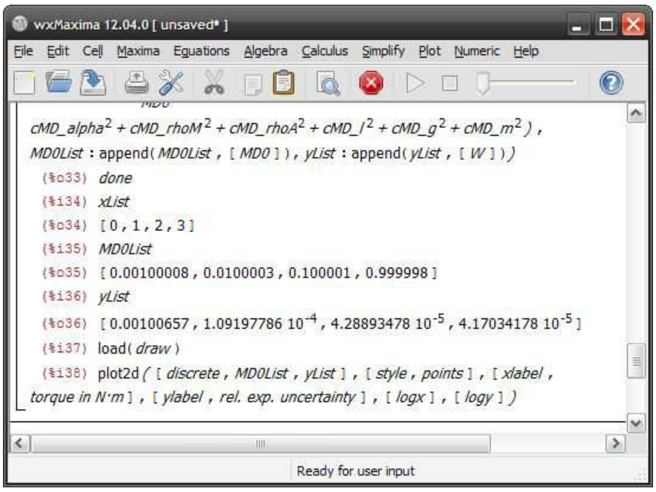

Figure 10: Screenshot from Maxima: calculated uncertainties *yList* of the torques generated in a torque standard machine in the range from 1 mN·m to 1 N·m.

A limitation of the tool at the moment is that only one quantity with a list of values can be handled. In the future it should be possible to have many such quantities and to calculate n-dimensional arrays of results and uncertainties. But the question of their graphical representation still needs to be answered.

6. CONCLUSIONS

The paper describes how the CAS Maxima can be used for calculating measurement uncertainties in the case that the model equation and the values of all input quantities including their uncertainties are known. Some further free software tools make the usage of this package quite easy even for less experienced users.

The procedures described here enable everybody with at least a little knowledge of programming to obtain the results in a shorter time and with less effort compared with fully featured software packages. The methods can also be used to prove the results obtained by other means with an independent calculation.

Although this paper cannot cover all aspects of uncertainty calculations, the author is optimistic that the tools discussed here may be helpful in many applications.

7. ACKNOWLEDEMENTS

The research leading to these results has received funding from the European Union on the basis of Decision No 912/2009/EC. The EMRP is jointly funded by the EMRP participating countries within EURAMET and the European Union.

EMRP
European Metrology Research Programme
▶ Programme of EURAMET
The EMRP is jointly funded by the EMRP participating countries within EURAMET and the European Union

8. REFERENCES

[1] Wikipedia: List of uncertainty propagation software, Web: http://en.wikipedia.org/wiki/List_of_uncertainty_propagation_software, Accessed: 26 June, 2013.

[2] Maxima, a Computer Algebra System, Web: http://maxima.sourceforge.net/, Accessed: 26 June, 2013.

[3] Guidelines on the Calibration of Static Torque Measuring Devices, EURAMET cg-14, Version 2.0 (03/2011), Web: http://www.euramet.org/fileadmin/docs/Publications/calguides/EURAMET_cg-14__v_2.0_Static_Torque_Measuring_Devices_01.pdf, Accessed: 26 June, 2013.

[4] Evaluation of measurement data – Guide to the expression of uncertainty in measurement, JCGM 100:2008, Web: http://www.bipm.org/utils/common/documents/jcgm/JCGM_100_2008_E.pdf, Accessed 15 June, 2013.

[5] Notepad++, a free source code editor, Web: http://notepad-plus-plus.org/, Accessed: 30 September, 2013.

[6] AutoIt, a freeware BASIC-like automation and scripting language, Web: http://www.autoitscript.com/site/, Accessed: 30 September, 2013.

[7] D. Röske: Uncertainty Contribution in the Case of Cosine Function with Zero Estimate – a Proposal, Proc. of the IMEKO 2010 TC3, TC5 and TC22 Conferences Metrology in Modern Context, Pattaya, Chonburi, Thailand, November 22-25, 2010, Web: http://www.imeko.org/publications/tc3-2010/IMEKO-TC3-2010-028.pdf, Accessed: 30 September, 2013.

[8] Gnuplot - a portable command-line driven graphing utility for Linux, OS/2, MS Windows, OSX, VMS, and many other platforms, Web: http://www.gnuplot.info/, Accessed: 6 October, 2013.