Using a Free Open Source Software to Teach Mathematics

FRANCISCO BOTANA,[1] MIGUEL A. ABÁNADES,[2] and JESÚS ESCRIBANO.[3]

[1] Departamento de Matemática Aplicada I, Universidad de Vigo, Campus A Xunqueira, 36005 Pontevedra, Spain

[2] CES Felipe II, Universidad Complutense de Madrid, 28300 Aranjuez, Spain

[3] Facultad de Informática, Universidad Complutense de Madrid, 28040 Madrid, Spain

ABSTRACT: We present the experience of the authors teaching mathematics to freshmen engineering students with the help of the open source computer algebra system Sage. We describe some teaching resources and present an ad hoc distribution of Sage used by the authors.

Keywords: Computer Algebra Systems, Open Source, Sage

1. INTRODUCTION

A computer algebra system (CAS) is a math computer program capable of working symbolically as well as numerically. It does on a computer the manipulation that has traditionally been done with pencil and paper. For instance, it immediately process computational tasks such as obtaining derivatives, integrals or graphs of one and several variable functions. Early CAS, such as MACSYMA and REDUCE, required considerable computer power and had a complex syntax. However, hardware and software advances over recent years completely changed this situation. Modern programs have progressed enormously beyond simply being manipulators of complicated symbolic strings and are now complete mathematical environments with algebraic, numerical, graphical and programming facilities.

CAS can have a significant impact on the way mathematics is taught and applied [1, 2, 3]. In fact, they have already become an important tool for many engineering and technical professionals. There is hence a growing need to incorporate the use of CAS into the education of all kinds of engineers [4].

This process is being delayed by software accessibility problems, for teachers and students, due to the commercial character of the main available CAS. It is our belief that an effective implementation of the use of CAS in math education will be enhanced through the use of open source systems. This is particularly important in emerging countries where most pressing needs within and outside the education system, make it unthinkable to devote significant parts of the budget for mathematical software. Even in the developed world, current and likely future education budget constraints make it doubtful that some universities will be able to continue the payment of some software licenses.

In this paper, we show how one can use an open source CAS to create teaching scenarios similar to those developed by other authors using alternative systems (e.g. [5, 6, 7, 8, 9, 10]).

Open source software development, with its philosophy resembling a bazaar of different agendas and approaches, was thought at the beginning to be only good for small applications. Its success, clearly exemplified by the magnitude of the operating system GNU/Linux, came as a surprise to even the most optimistic programmers, who thought that big applications would always need a reverent cathedral building approach [11]. Nowadays, open source applications have impacted all computing areas and are no longer considered marginal.

If we take open as a synonym for accessible, we cannot find fields where this concept is more relevant than education, where universal access should be the leading principle, and mathematics, where public scrutiny is at its very core. In this paper we give notice of a relatively new free open source CAS illustrating its use in education.

In a research context, the closed nature of commercial CAS, whose code is not generally available for examination, raises reliability concerns. In a higher educational context, trusting the software is not so much the issue as it is teaching mathematics upon the same foundations as mathematics is built upon: openness.

The selected system, Sage [12], is an open source general purpose CAS whose philosophical foundation can be summarized as applying the system of open exchange and peer review characteristic of scientific discourse to the development of mathematical software.

2. SOME FEATURES OF SAGE

Sage was initiated in 2004 by William Stein (http://wstein.org) as a project to develop a free open source CAS that any teacher, researcher or student could use freely (in a wide sense, including access to the code) and, at the same time, a CAS scientifically rigorous, in the sense that all the algorithms and methods could be checked and improved by anyone. Sage is in constant development. What started as an interface connecting some already existing open source applications (Maxima, Singular, GNUplot,…), after developing a substantial amount of code specifically for Sage (by more than 180 developers) has become a core library bundling the functionality of the different components into one consistent experience [13]. At the moment, besides the main system, there exist around 100 packages used in Sage [14].

In its web page http://sagemath.org, one can read that the mission of Sage is "*Creating a viable free open source alternative to Magma, Maple, Mathematica and Matlab*". Is Sage performance similar to that of other standard CAS? This is a difficult question to answer. All these systems are huge applications, with hundreds of commands and modules, and, depending on what you are interested in, the performance of each system can be different. A general technical study comparing the different CAS is a work to be done, but it is out of the scope of this paper. However, we can mention that some studies have been done, mainly in number theory, determining that Sage is really fast performing some difficult calculations [15, 16].

Another main characteristic of Sage is the use of Python as base language. Python is an object oriented language, easy to learn and widely used. Most CAS provide their own programming language, while Sage developers have chosen an existing one. Hence, a user that learns how to use Sage, will also be learning this quite popular scripting language. And, if you already know Python, learning basic Sage programming will be an easy process. Of course, to learn mathematics is not necessary to know Python. But, for a student, to have a rigorous mathematical education together with programming skills in a widely used language such as Python is very important in order to obtain a solid professional career.

Sage has a command line interface, but the easiest way to interact with Sage is through its graphical interface: *notebook*. Once you launch the "notebook()" command, you can access the graphical interface using any standard web browser. In fact, you can use Sage like a web server, locally or, with a simple configuration, like a remote web server. This is very useful to access your work from any place or to share your work with, for example, students. If you do not want to install your own Sage server, you can use a public one, like http://www.sagenb.org, where you can try Sage without installing anything. All your work is saved in what is called a "Sage worksheet".

Installing Sage is easy, but we must consider some details. As mentioned before, we can install Sage locally or we can use a Sage server. Sage installation on GNU/Linux, Unix or Mac is immediate. To work with Sage in a Windows system one has to install a virtual machine like VirtualBox. It must be highlighted that the company Microsoft is funding the Sage-Windows portability [17]. But the simplest (for a student oriented use) way to use Sage is through a Sage server: just connect to a server and use it!

Despite the large number of users, the www.sagenb.org server shows a surprisingly high connection speed. This makes it completely reliable as the main tool during a lab class session, for instance. And there are other servers available for public use as well [18], but if a generalized use of Sage is going to be done in a class or institution, we recommend to install and configure your own Sage server to have absolute control over the system.

From its original design, it is easy to integrate external modules or applications in a Sage worksheet. Since a worksheet is basically a web page you can integrate, for example, a Java applet. This allows the use of external resources written in Java. For example, you can launch the dynamic geometry software GeoGebra [19] within your Sage worksheet. Sage has also specific interfaces to interact with well known applications, like TeX/LaTeX, Singular,… even with commercial CAS like Mathematica, Maple or Matlab [20]. Hence, if you have been working with these CAS for a long time, and you have written materials in these formats, the transition to Sage using these interfaces is an easy task. For instance, suppose you have written a long algorithm, *MyLimit*, in Mathematica for computing, say, the limit of a scalar field. This code can be reused from inside Sage if you have Mathematica in your computer. Or, if you do not remember how to factorize in an algebraic extension with Sage, you can use Mathematica as shown by the following computation

```
mathematica('Factor[35x^2-y^2-210x+315,Extension->Sqrt[35]]')
    -((105-35*x+Sqrt[35]*y)*(-105+35*x+Sqrt[35]*y))/35
```

3. TEACHING APPLICATIONS

The following are some teaching applications of Sage illustrating some of the different characteristics pointed out above.

3.1. Sage as an all-purpose Toolbox for Advanced Calculus

The use of CAS in engineering education has not resulted (yet) in the revolution that some predicted in the early 1980's (see [21]). Being CAS ultimately symbolic calculators, an impact of their use in university level education similar to that of hand-held calculators in high schools was expected. Lack of wide access of students to CAS has been pointed out as one of the main reasons for this not being the case [22]. The open source and web based nature of Sage can help solve the accessibility problem.

In this section we focus on the use of Sage as an advanced calculator with some illustrative examples developed under the leading idea that interactive learning allows students to discover the principles by themselves, making technology an efficient learning resource. The use of *real-world* technologies during their learning process is also a perfect mean to acquire the technical attitudes and skills required to tackle a problem successfully, so important later in a professional career.

Much like standard hand-held calculators, the use of CAS allows to free students from tedious routine computations by hand. This opens the door for a deeper learning process in which the emphasis is put on mathematical content. Instead of practicing artificially complex integration techniques on the few trivial examples that work, it is possible to further explore the meaning of integration and its numerical approximation. And the same applies for much of the standard calculus and algebra curricula. Students can be invited to interactively experiment with more realistic problems, problems whose solution is not unique and given by one exact formula. In summary, students learn what real Mathematics are about.

These ideas have been traditionally followed (when followed) as a separate part of a math course: in special sessions in separate labs and in special days of the semester. Now that the use of portable devices (netbooks, tablets, smartphones,…) by the students is generalized, the internet accessible Sage notebook makes it possible to really incorporate its use as part of any math class. Although there are efficient online tools that can be used as symbolic calculators (Wolfram|Alpha, for instance), the notebook applications can be tailored to specific learning needs for students and interests of teachers.

In Sage, it is easy to develop interactive templates for particular exercises using the *interact* command. We call these templates interacts (see [23] for a list of them). This makes the use of Sage more friendly, something that is important especially to first year college students. The authors have developed a live DVD of Sage that has been distributed to first year students of the forestry engineering school of the University of Vigo, Spain. This distribution includes thirty Sage interacts that have been used as complementary material in a course covering the standard calculus sequence for engineering students.

In this course, students are invited to bring their laptops to class from day one. They are provided with a copy of the live DVD (or USB live for those who do not have the appropriate reader). Students who use Windows (95% in the two semesters that this distribution has been used) are invited to install a partition on their hard drive for better performance. Those who have no computer in the classroom are offered an account for a Sage server with restricted access, where all teaching material is available for use outside the classroom. We have observed that about half of the students usually take their laptops to classroom, and they have no problem sharing them with other students. This way, digitally skilled students help the less-skilled ones and the learning of the user interface is very fast.

Students use the interacts to solve the proposed assignments (which can be handed back as Sage worksheets), and in general, as a symbolic calculator to solve problems. We found no noteworthy problems regarding the effective use of the interacts, beyond system installation problems on older computers and the initial need for learning some basic syntactic rules.

As noted above, using these interacts students save time by avoiding tedious and uninteresting mechanical computations, as reported by most students. The student can hence concentrate on concepts rather than on computational skills as recommended by some authors [24].

These Sage interactive worksheets are the Sage equivalent to some webMathematica interactive resources used by the authors in previous years. The migration to Sage was in part forced by budgetary limitations. All the material (in Galician, a co-official language in Spain) relative to these resources is freely available at [25].

In Figure 1 an interact that provides the graph of a gradient vector field is shown. It is worth mention that this 3D graph can be interactively rotated.

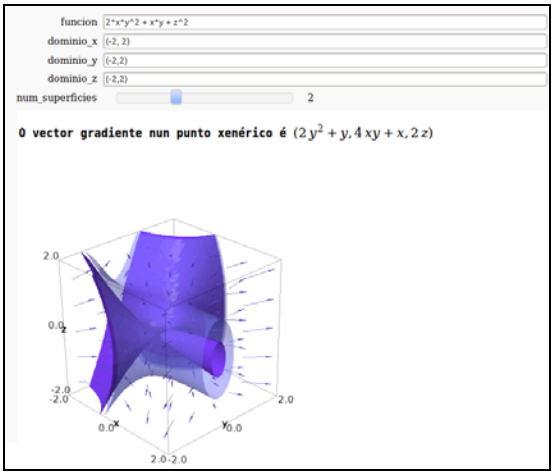

Fig. 1: Gradient field of a multivariate function.

Let us emphasize that Sage, when using the interacts or as a web application, lets the user access its code, allowing its modification and personalization. We consider this non-authoritarian learning experience one of the main advantages of Sage versus other systems.

3.2. Sage as an Efficient Symbolic Complement for Applets

In this section we sketch one important feature of the Sage notebook: the easy integration of applets in Sage worksheets. We illustrate this useful characteristic with two examples dealing both with classical Calculus topics, namely Lagrange multipliers and vector fields.

In the first example we put together in a Sage worksheet an applet that numerically computes extrema of a function and a Sage interact providing the exact computation following the symbolic approach.

In [26] an applet for approximate computing of Lagrange multipliers for functions of two variables is offered. Reading the text form of the web page we can find the URL of each applet file and then use it embedded in the worksheet, simply by using the following html code:

> *html('<applet code="LagrangeMultipliersTwoVariables" archive="http: //ocw.mit.edu/ans7870/18/18.02/f07/tools/lagrangeMultipliersTwoVari ables.jar, http://ocw.mit.edu/ans7870/18/18.02/f07/tools/mk_lib.jar , http://ocw.mit.edu/ans7870/18/18.02/f07/tools/parser_math.jar, ht tp://ocw.mit.edu/ans7870/18/18.02/f07/tools/jcbwt363.jar" width=760 height=450></applet>')*

Adding a simple code for the symbolic computation of the extrema candidates, the worksheet (Figure 2) facilitates both an interactive graphic and an interactive symbolic approach to learning about this topic. Nevertheless, again, the mathematical code behind the symbolic part remains accessible for examination or modification.

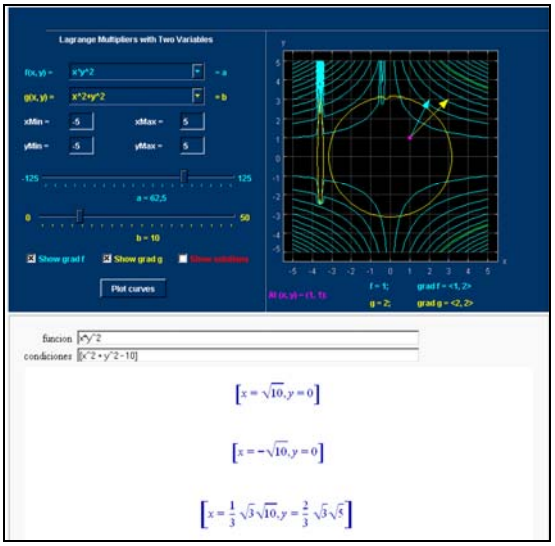

Fig. 2: Lagrange worksheet showing the applet and part of the symbolic solutions.

The interested reader can download the worksheet, called "Lagrange-multipliers-applet-interact", developed by the authors, in [27], and experiment with it. Note that although the applet is fully functional, a user must be logged in to obtain a symbolic result.

While the applet in the example above provides a nice illustration to accompany the Sage interact that symbolically computes the extrema of a function, there is no direct communication between the applet and the Sage interact. In the second example, we show how in some situations, Sage and the applet can share information, allowing for further applications.

In Figure 3 a Sage interact with a geometric applet is shown. The three functions entered in the interact automatically define the curve and the vector field shown in the applet. The applet, generated by Cinderella [28], shows the animated vector field, its value *f* for the draggable point *A* and the tangent vector to the curve at the point *A* (when *A* is on the curve).

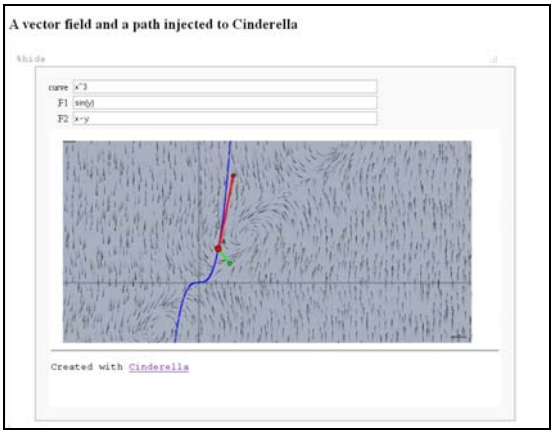

Fig. 3: Sage interact with a Cinderella applet.

This interact is more a proof-of-concept than a finished teaching resource. Our aim here is to illustrate the connection possibilities between the Sage notebook and a well established interactive geometry program. The interested reader can download the worksheet, called “Cinderella Interact”, in [29].

3.3. Sage as a Dynamic Geometric Experimentation Environment

The name of Dynamic Geometry Systems (DGS) is given to the family of computer applications that allow exact on-screen drawing of (generally) planar geometric diagrams and their interactive manipulation by mouse dragging certain basic elements. Besides Cinderella, special mention deserves the system GeoGebra, whose open source model and effective community development has resulted in a spectacular worldwide distribution that basically makes it a de facto standard in the field.

Although most DGS come equipped with some property checker, their numeric nature makes the answers unreliable. The idea behind this example is the interconnection of Sage and GeoGebra to compensate the computational limitations of the latter.

The prototype presented here consists of a Sage worksheet in which two different tasks are performed over GeoGebra constructions. More precisely, automatic deduction techniques based on Groebner bases are used to either compute the equation of a geometric locus in the case of a locus construction or to determine the truth of a general geometric statement included in the GeoGebra construction as a Boolean variable. The Sage worksheet includes a GeoGebra applet that allows the direct construction of a

diagram or the upload of a local previously designed GeoGebra construction. Let us remark that both tasks are performed symbolically, providing *certified* answers.

Its main technical characteristic, and the one that makes the user experience completely automatic, is the intensive use of JavaScript to allow the direct communication between Sage and the GeoGebra applet. This has made possible to circumvent the question/answer nature of Sage to generate what amounts to a one-click add-on for GeoGebra.

We illustrate the possible teaching scenarios based on this system with two examples: the proof of a basic theorem in the form of a GeoGebra construction with a Boolean variable and the computation of the equation of a classical locus. In both cases the idea is to complete the sequence construct-experiment-conjecture with a completely reliable answer. The use of tools like the one presented here also helps an inexperienced student understand the difference in reliability between a numerical computation and a symbolic answer based on deep algebraic structures.

We consider the basic theorem that states that the three altitudes of a triangle meet in one point, its orthocenter. GeoGebra can check that the altitudes have a single common point, as illustrated in Figure 4. This result can be easily reproduced in terms of a boolean variable in a GeoGebra construction. Given triangle *ABC*, it suffices to consider the intersection point *P* of the altitudes *a* and *b* (through vertices *A* and *B* respectively). The Boolean statement encompassing the theorem is "line(*C*,*P*) perpendicular line(*A*,*B*)?". The answer (true) provided by GeoGebra, based on numerical computations, is symbolically corroborated by Sage when returning "The GeoGebra statement is generally True".

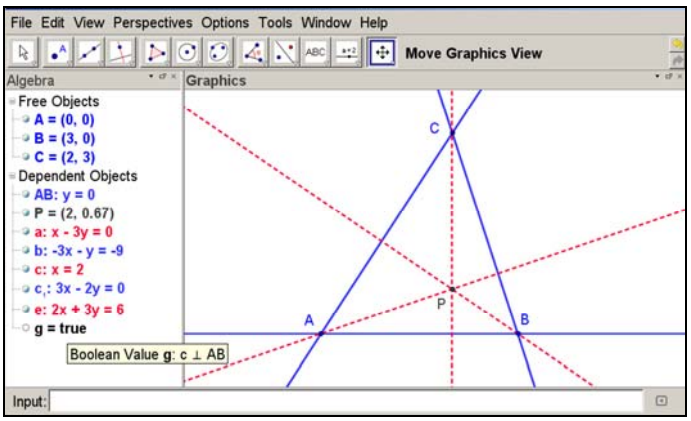

Fig. 4: Numerically checked answer provided by GeoGebra.

Notice that, unlike the answer provided by GeoGebra, the answer provided by Sage is completely general. Symbolic variables are internally used as generic coordinates for the three vertices, what makes the answer a general statement about any generic triangle.

In the second example we consider the construction of the classical conchoid of Nichomedes. In Figure 5 (up) we can see how GeoGebra plots only part of the curve in some instances.

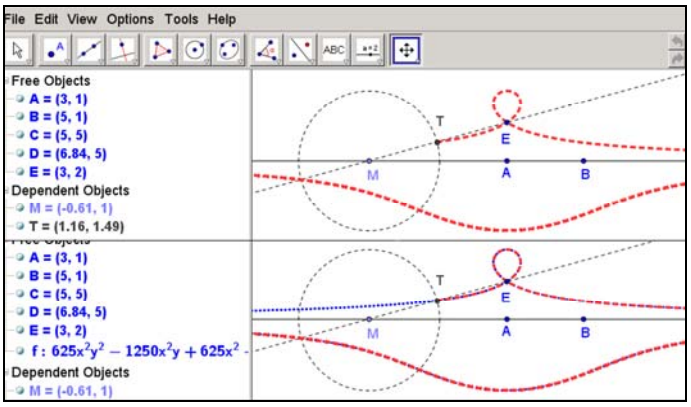

Fig. 5: Locus graph provided by GeoGebra (up) and graph completed by Sage (down).

Apart from the incomplete GeoGebra graph, the found conchoid is just a graphic object, while Sage computes its exact algebraic description, the quartic

$$625x^2y^2+625x^2+625y^4-1250x^2y-3750xy^2-3750y^3+$$

$$+7500xy+11634y^2-3750x-10286y-339=0$$

which is injected in the GeoGebra applet as shown in Figure 5 (down).

These examples show again how the easy integration of Java applets in Sage make it a suitable symbolic companion for any DGS that allows exporting constructions as applets, as it is the case with most systems.

The interested reader can download the worksheet, called “GeoGebra-locus-proof”, developed by the authors, in [30], and experiment with it once logged in to the server.

3.4. Sage as a Remote Math Server. The Flexible Ladder Problem

In sections 3.2 and 3.3 above, it was shown how one can enrich a Sage worksheet by inserting Java applets. In particular, in section 3.3 a GeoGebra applet was used to solve automatic deduction tasks. Now we show how one can somehow reverse that approach and bring Sage to the applet instead of the applet to Sage. More precisely, we show one example of web-based teaching resource in which a GeoGebra applet is complemented with access to Sage, allowing the automatic solving of involved geometric problems not possible in standard dynamic geometry systems. The resource offers the general framework for a particular geometric question in which the main element can be changed by the user. For this reason, we refer to it as a *template*. It is worth mentioning that the development of this template, besides access to a Sage server, only requires basic knowledge of JavaScript and HTML.

In the sliding ladder problem the student is asked to find the shape of the curve described by a point of a ladder when sliding to the floor from its position leaning against a vertical wall [31]. The template considers a recent generalization of this problem in which the ladder is allowed to have variable length [32]. The template proposes the problem of finding the envelope of the family of lines determined by the different positions of a flexible ladder defined by the segment joining the origin and a point in the graph of a general function.

More precisely, besides a GeoGebra applet in which the flexible sliding ladder is constructed, the template includes a text area and three text fields accompanied by instructions. Although we can retrieve and show in a text area the information provided by the remote Sage server, for security reasons we cannot have direct access to these data from the web page.

The three text fields are used to "talk" with Sage and the applet. First there is a technical field in which the user has to paste the Sage session number as provided in the text area. This is necessary for subsequent requests to the server. In the second text field, the user has to indicate the function describing the flexible ladder. Once this is done, the user just has to press a button for Sage to provide the equation of the sought envelope in the text area. Once the equation is shown in the text area, a process that takes a few seconds, one finally has to copy/paste it in the last text field for its graph to be displayed in the GeoGebra applet.

Behind all this seemingly simple process there are of course involved computations remotely performed by Sage (Singular in particular).

As an example, the envelope for the family of lines (in Figure 6, left) determined by the function $f(x)=(x-1)^3-3(x-1)^2+5$ is given by the following equation of degree 6:

$$135x^6+432x^5y+72x^4y^2+4x^3y^3-1620x^5-3942x^4y-666x^3y^2-36x^2y^3+7290x^4+$$
$$+13176x^3y+5031x^2y^2+594xy^3+27y^4-14310x^3-15606x^2y-2646xy^2-216y^3+$$
$$+9315x^2-378xy+486y^2+2430x-432y+135 = 0$$

whose graph is shown in Figure 6 (right) together with the graph defining the flexible ladder.

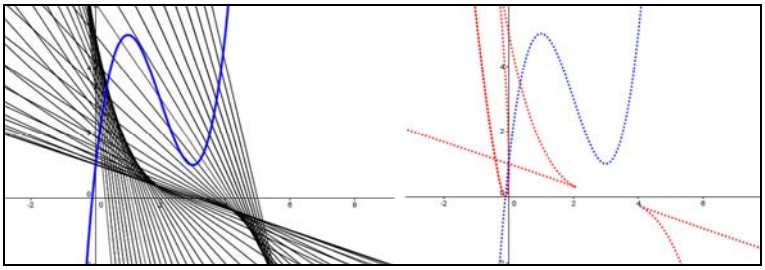

Fig. 6: Family of lines determined by a flexible sliding ladder (left) and its envelope (right).

The key point of this example is to show how it is possible to make available the computational capabilities of Sage to a GeoGebra applet within a simple web page. This is done interacting with a Sage session over HTTP through what is known as a *simple Sage server*. Notice that this allows a teacher to emulate this template without the effort of maintaining a Sage server by remotely using a Sage server; ideally one provided by his/her educational institution. The interested reader can access this template, developed by the authors, in [33] and experiment with it.

3.5. Sage as a Platform to Share

As illustrated by the examples above, one can create a wide variety of teaching applications using Sage. However, creating a meaningful worksheet can be time consuming. This process can be drastically simplified by using one of the hundreds of public worksheets available at the main server [34]. The worksheets, published and ranked by the users can be downloaded to be used in their original form or to be modified to better suit our teaching goals.

Good examples of the many teaching materials that can be obtained from this source are the worksheets *Graphs of Elementary Numerical Integrals* [35], *radian measure* [36] and *Fourier* [37].

CONCLUSIONS

CAS will inevitably become as standard a tool for doing mathematics for students and engineers at universities as the hand-held calculator is today in high schools. Students will use CAS to solve their mathematical everyday problems and teachers will use the power of CAS to enrich their lectures with more nontrivial mathematical content.

However the process of introducing CAS in the classroom is being slowed by the lack of universal access to the software, for students and for teachers. We have shown how the free open source CAS Sage is a real teaching alternative to proprietary CAS. It eliminates the accessibility problem while providing state of the art computational capabilities.

ACKNOWLEDGEMENTS

We thank the referees for all their valuable remarks and suggestions. This work has been supported by the Spanish MINECO under grants MTM2008-04699-C03-03 and MTM2011-25816-C02-02.

REFERENCES

[1] R. Pierce, R. and K. Stacey, Reflections on the Changing Pedagogical use of Computer Algebra Systems: Assistance for Doing or Learning Mathematics?, *J. Comput. Math. Sci. Teaching*, Vol. 20, No. 2, 2001, pp. 143-161.
[2] C. Hoyles and J. B. Lagrange, *Mathematics Education and Technology - Rethinking the Terrain. The 17th ICMI Study*. Springer, New York, 2010.
[3] B. Kramarski and C. Hirsch, Using computer algebra systems in mathematical classrooms. *J. Comput. Assist. Learn.*, Vol. 19, No. 1, 2003, pp. 35-45.
[4] G. Barozzi and R. R. Clements, The potential uses of computer algebra systems in the mathematical education of engineers, *Int. J. Math. Educ. Sci. Tech.*, Vol. 18, No. 5, 1987, pp. 681-683.
[5] F. Carneiro, C. P. Leão and F. C. F. Teixeira, Teaching differential equations in different environments: A first approach, *Comput. Appl. Eng. Educ.*, Vol. 18, No. 3, 2010, pp. 555-562.
[6] G. Jovanovic Dolecek, Interactive MATLAB-based demo program for sum of independent random variables, *Comput. Appl. Eng. Educ*, doi: 10.1002/cae.20481.
[7] A. Elkamel, F. H. Bellamine and V. R. Subramanian, Computer facilitated generalized coordinate transformations of partial differential equations with engineering applications, *Comput. Appl. Eng. Educ.*, Vol. 19, No. 2, 2011, pp. 365-376.
[8] E. A. Cariaga and M. C. Nualart, Teaching and learning iterative methods for solving linear systems using symbolic and numeric software, *Comput. Appl. Eng. Educ.*, Vol. 10, No. 2, 2002, pp. 51-58.
[9] Y. Jiang and C. Wang, On teaching finite element method in plasticity with Mathematica, *Comput. Appl. Eng. Educ.*, Vol. 16, No. 3, 2008, pp. 233-242.
[10] J. T. Chen, K. S. Chou and S. K. Kao, 2009, One-dimensional wave animation using Mathematica, *Comput. Appl. Eng. Educ.*, Vol. 17, No. 3, 2009, pp.323-339.
[11] E. S. Raymond, The Cathedral and the Bazaar, *Knowl. Technol. Polic.*, Vol. 12, No. 3, 1999, pp. 23-49.
[12] W. Stein and D. Joyner, SAGE: System for algebra and geometry experimentation. *SIGSAM Bull.*, Vol. 39, No. 2, 2005, pp. 61-64.
[13] Press kit, http://www.sagemath.org/library-press.html (last accessed April 2012)
[14] Sage components, http://www.sagemath.org/links-components.html (last accessed April 2012).
[15] Sage benchmarks, http://www.sagemath.org/tour-benchmarks.html (last accessed April 2012).
[16] Sage comparisons, http://wiki.sagemath.org/Comparisons (last accessed April 2012).
[17] The Sage mathematical software project, http://research.microsoft.com/apps/video/dl.aspx?id=103515 (last accessed April 2012).
[18] Sage notebook, http://wiki.sagemath.org/sagenb (last accessed April 2012).
[19] GeoGebra, http://www.geogebra.org (last accessed April 2012).
[20] Sage interfaces, http://www.sagemath.org/doc/reference/sage/interfaces (last accessed April 2012).
[21] L. A. Stern, Computer calculus, *SIGSAM Bull.*, Vol. 15, No. 3, 1981, pp. 26-27.
[22] D. Lawson, The challenge of computer algebra to engineering mathematics, *Eng. Sci. Educ. J.*, Vol. 6, No. 6, 1997, pp. 228-232.
[23] Interact – Sage Wiki, http://wiki.sagemath.org/interact/ (last accessed April 2012).
[24] B. Buchberger, Should Students Learn Integration Rules?, *SIGSAM Bull.*,Vol. 24, No. 1, 1990, pp. 10-17.

[25] PPW2.0 Prácticas de matemáticas Pola Web, http://webs.uvigo.es/fbotana/ppw20 (last accessed April 2012).
[26] Lagrange multipliers applet, http://ocw.mit.edu/ans7870/18/18.02/f07/tools/LagrangeMultipliersTwoVariables.html (last accessed April 2012).
[27] Lagrange multipliers interact, http://alpha.sagenb.org/home/pub/168 (last accessed April 2012).
[28] J. Richter-Gebert and U. Kortenkamp, *The Cinderella.2 Manual*. Springer, Berlin, 2012.
[29] Cinderella Interact worksheet, http://alpha.sagenb.org/home/pub/368 (last accessed April 2012).
[30] GeoGebra locus-proof worksheet, http://alpha.sagenb.org/home/pub/169 (last accessed April 2012).
[31] Ladder problems, http://www.mathematische-basteleien.de/ladder.htm (last accessed April 2012).
[32] T. M. Apostol and M. A. Mnatsakanian, A New Look at the So-Called Trammel of Archimedes, *Am. Math. Month.*, Vol. 116, No. 2, 2009, pp. 115-133.
[33] Flexible ladder template, http://nash.sip.ucm.es/Ggb-directPlus/Ggb-directPlus-template-flexible-ladder.html (last accessed April 2012).
[34] Published Sage worksheets, http://www.sagenb.org/pub/ (last accessed April 2012).
[35] Numerical integrals worksheet, http://www.sagenb.org/home/pub/2650/ (last accessed April 2012).
[36] Radian measure worksheet, http://www.sagenb.org/home/pub/1469/ (last accessed April 2012).
[37] Fourier worksheet, http://www.sagenb.org/home/pub/4575/ (last accessed April 2012).