数理解析研究所講究録
第1909巻 2014年 184-199

FunMath-MPSv0.1: A *New Mathematica Package Suite* for Optimizating the Learning Process in "Fundamentals of Mathematics" Course

Andrés Iglesias[1,2], Robert Ipanaqué[3]
[1]*Department of Applied Mathematics and Comp. Sciences*
University of Cantabria, Avda. de los Castros
s/n, E-39005, Santander, SPAIN
[2]*Department of Information Science*
Faculty of Sciences, Toho University
2-2-1 Miyama, 274-8510, Funabashi, JAPAN
E-mail: iglesias@unican.es
Web site: http://personales.unican.es/iglesias
[3]*Department of Mathematics, National University of Piura*
Urb. Miraflores s/n, Castilla, Piura, PERU
E-mail: robertchero@hotmail.com
Web site: http://www.unp.edu.pe/pers/ripanaque/

Abstract

This paper corresponds to the printed form of an invited talk the first author delivered during a RIMS workshop held at Kyoto University in August 2013. The presentation was mostly devoted to show a preview of a new mathematical software developed by the authors, preliminary called *FunMath-MPS* version 0.1. The software is basically an on-going project towards the development of a full suite of Mathematica packages providing support to the computational needs of both teachers and students of the *"Fundamentals of Mathematics"* course, which is traditionally taught in Mathematics, Physics, and other Engineering degrees in many Universities and Colleges all over the world. This paper explores some issues regarding the development of this suite. It also shows some preliminary examples of some on-going packages in different subjects: graphical representation and algebraic manipulation of piecewise functions, and row and column operations on matrices.

1 Introduction

On August 21st. 2013, the first author was kindly invited to deliver a talk at the *"RIMS Workshop: Study of Mathematical Software and Its Effective Use for Mathematics Education"*, held at the Research Institute for Mathematical Sciences (RIMS) of Kyoto University, Kyoto, Japan (`http://www.kurims.kyoto-u.ac.jp/en/index.html`). The primary goal of this workshop was to analyze the interplay between mathematical software (mostly CAS, computer algebra systems) and education regarding the effective use of mathematical software in order to improve the educational level and bring out the best of teachers and students in all educational steps.

In that invited talk, we focused on the development of a new software program intended to provide support to the computational needs of both teachers and students of the *"Fundamentals of Mathematics"* course. This course is traditionally taught in Mathematics, Physics, and other Engineering degrees in many Universities and Colleges all over the world. The authors have identified more than 10,000 different study programs at university level including this subject into their curricula. Clearly, it is a subject of great interest for almost all scientific disciplines. It is also a challenging subject, one that is even hard to define. When asking teachers and students on what is this subject about, the classical answer is given in negative terms: it is simpler to describe what this subject is *not* rather than what is it. Common knowledge said that this course is neither Calculus nor Algebra nor Numerical Analysis, but some way it encompasses all them. Typically, the course uses a wealth of mathematical resources and concepts such as functions, continuity, derivation, integration, vector and matrix manipulation, equation and equation system solving (even inequalities in its broadest version), and graphical representation of functions (curve sketching, 2D, 3D). Very often, other additional subjects are added according to specific needs and students' profile. Those subjects range from Taylor series, numerical series, Laplace transforms, Fourier analysis, complex numbers and functions, optimization, numerical analysis, ordinary differential equations, and so on.

In addition to all those issues, our students have to meet the new challenges of an increasingly technological world. This challenge is specially important in very difficult subjects such as Mathematics and other scientific disciplines, which typically suffer from high failure rates. Several academic reports have pointed out the difficulties our students face when studying (and suffering) mathematical subjects. Students and professors at university often cite lack of preparation from high school, poor study habits, and the rapid pace of the course as reasons for such low scores, but it is clear that there are many reasons happening simultaneously to explain why these problems arise recurrently everywhere.

To make things even worse, the profile of our current students is also challenging, unbalanced, and troublesome. In general, they are less skilled than their counterparts in the last decades in deduction, mathematical intuition, and scientific reasoning and encounter more problems in solving questions with scientific content. Their background is also less solid in both science and arts. Furthermore, they also have less oral and written communication skills, with a much limited vocabulary and, hence, find some

troubles for a full comprehension of concepts and ideas. Very often, they lack discipline and exhibit poor study habits such as poor note-taking skills, poor time management, last minute work, procrastination, over-reliance on classmates and/or Internet, and so on. On the positive side, most current students come to college and university with greater computer proficiency and technology skills than their predecessors.

In our opinion, part of the solution to these problems must come from a good collection of supporting materials, of which computer software is undoubtedly a primary resource. Mathematics is very much practice-based. Students may grasp a concept in the classroom, but they will certainly lose it if not reinforced by homework. For doing so, students need to be provided with good supporting materials so that they can effectively learn by themselves. High-quality, helpful educational materials allow underachieving students catching up on belated assignments and get extra time for successful backtracking. It is at this point where Computer Algebra Systems (CAS onwards) can really pave the way, making the most with less. In author's opinion, CAS are very powerful tools (arguably the best ones) to face the challenges of this new approach to Higher Education.

Fortunately, there is a wealth of computational software coming in our help. Among them, the symbolic computational programs have already proved to be very effective tools in order to improve the general performance of our students both at the classroom and for homework. There is a limiting factor, however, in this educational approach. The most popular symbolic tools so far are commercial software and, in general, tend to be costly for standard students. In fact, we have witnessed a big rise in the cost of this symbolic software. Even although the companies developing this kind of software have tried to create special client license programs in order to increase their customer base and allow more people to access their programs, it is clear that economical issues have become increasingly important.

The other factor is that commercial mathematical software is mostly oriented to professionals and educators, and much less emphasis is put upon their use by students. As a consequence, the majority of mathematical software packages assume a certain level of mathematical knowledge that is not commonly acquired yet by freshmen and sophomore university students. Clearly, there is gap between the basic knowledge required by those mathematical packages and that owned by our current students, even at university level.

Most teachers overcome this gap by creating a series of course materials by themselves in order to support the educational needs of their students. No need to mention, creating this *"customized"* material is a very time-consuming task. Most teachers claim they have not enough time to develop this *"à la carte"* material on a regular basis. Our aim is to help them by creating such educational-oriented material and embed it into a collection of packages for different subjects typically included in the *"Fundamentals of Mathematics"* course.

Next sections will describe our approach in some detail.

2 Software Design

This section describes the fundamental steps towards the development of the mathematical software introduced in previous section. Main steps of this process are the conceptual design and the software architecture. They are discussed in next paragraphs.

2.1 Conceptual design

First main requirement of any proposal aimed at achieving the goals described in previous section is *modularity*. Our contents should have a strong modular character. This requirement is given by the fact that the course is extremely general and varied in terms of its constituent subjects. The only way to adapt to very different syllabus, students' profile, and goals is to organize the material as a series of independent modules, kind of bricks in a wall. This modular structure allows teachers to select the modules involved in the subject they are teaching in their own course, regardless how the other subjects have been created. It also allows students to surf freely into the material without having to follow a sequential, inter-dependent approach.

Second main requirement concerns the *usability*. This a vague concept but what lies behind is the idea of ease of use, so that any student and educator should grasp how to use the software with a minimal time and effort. To this aim, we promote the extensive use of palettes, buttons, menus, and other easy-to-use, user-friendly interactive tools. Very often, a nice graphical user interface is the preferred way to operate, because it maximizes the use of user-friendly tools and resources. This is also the approach we intend to follow for our system.

Third main requirement is the *extensibility*. Given the highly dynamic character of the subjects included in this course, it is important to ensure that the system is designed in such a way that new modules and extensions can readily be added, much in a plug-and-play way. The basic idea is that any teacher or student can develop his/her own program about a specific subject and plug it into the system. The software should accept this new plug-in and operate with it in a standard way (similar to the built-in original code of the packages), provided that a specific format is met. Of course, this requires the development of a standardized format so that all new plug-ins can be smoothly developed and deployed into the main core.

2.2 Software architecture

Once the main conceptual design requirements of our system are determined, next step concerns its implementation. Given its educational approach towards mathematical contents at university level, we obviously need to combine the expressive power of symbolic computation with the speed of numerical computation. On the other hand, we need an extensible software with a powerful programming language and a nice, user-friendly graphical user interface. Given those requirements, it came to our minds that *Mathematica* is specially praised for those appealing features. *Mathemat-*

ica is a very popular computer algebra system with outstanding symbolic capabilities, a powerful programming language well equipped with pattern-recognition and functional programming features, a very nice and intuitive graphical interface, an extensive gallery of document stylesheets and many other valuable functionalities (such as support for complex numbers, arbitrary precision, animation tools, procedural, functional and object-oriented programming capabilities, support for a bulk of import/export formats, connection tools to DLLs, Java, C++, Fortran, CUDA, LaTeX, OpenCL, powerful technical word processing including mathematical editing (such as formulas and expressions), high-performance computing tools, automatic muti-threading, and so on. On the other hand, the distinction between the computing core (called the *kernel*) and the graphical interface (called the *front end*) simplifies many software developments tasks and the implementation pipeline, which at turn implies a significant reduction of the software development life cycle. On the other hand, the computing core is enriched by a collection of libraries (*packages* in *Mathematica*'s terminology) that can be invoked by the user at any time of a running session and extend *Mathematica* core capabilities by providing extra functionalities for specific purposes. Given our main requirement for modularity, this is clearly the solution we were looking for. So, our system will consist of a collection (we called it *suite*) of *Mathematica* packages designed for each specific task.

A major difference between the *Mathematica* packages and the kernel is that the former are developed in the own *Mathematica* programming language, while the latter is compiled and hence, not available for end users. As a consequence, we are constrained by the fact that any new version of the kernel could potentially include new functionalities leading to a different behavior of the code in our suite. Fortunately, *Mathematica* provides several means to overwrite a given command or definition, so the code can be adapted to particular situations. On the other hand, *Mathematica*'s programming language is very powerful, and it includes symbolic computation, procedural programming, functional programming, rule-based programming, pattern-based programming, and term-rewriting programming, all in one (it is a genuinely called *multi-paradigm programming language*).

Our approach is to create a set of *Mathematica* packages devoted to the different topics susceptible of study in a "Fundamentals of Mathematics" course for Science and Engineering at university level. To this aim, each different topic is programmed in a different file (package). Then, different but related topics are enclosed in the same folder. The suite for the course consists of the collection of all those folders. Typically, there will be extra files associated with each folder for the definitions that have to be overwritten in order to avoid conflicts with the *Mathematica* kernel. Some other features (such as help files, examples, user's manuals and the like) can also be stored in the folders if needed.

3 Some Illustrative Examples

In this section we show some illustrative examples that were also shown and executed during the invited talk. We remark that the suite is still an on-going project, so consider current output and options as merely tentative. Therefore, the commands shown here are just for illustrative purposes and not necessarily reflect the final form, structure, and functionalities they will have for the time of the release of the suite. Probably, several changes will be made on the current version of our packages. Yet, we think these examples illustrate pretty well how do the packages work. In all examples described in next sections, we will assume we have already loaded the corresponding packages.

3.1 Piecewise functions

Manipulation of piecewise functions is usually tricky for mathematics students. Most computational tools do not help in this problem, as they do not provide features for the analysis of such functions in full detail. In next paragraphs, we describe some of those features and the solutions provided by our packages.

3.1.1 Graphical representation

Regarding the graphical representation of piecewise functions, think about the `Floor` function, which accepts a real number x and returns the greatest integer less than or equal to x. This is a piecewise function with a graphical representation:

```
Plot[Floor[x], {x, -3, 3}]
```

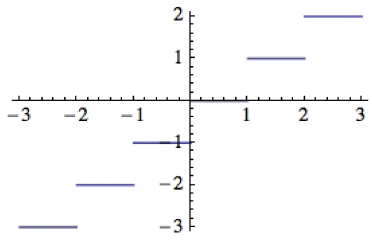

Figure 1: Graphical representation of the `Floor` function on the interval $[-3, 3]$.

Note that the end points of each interval are not properly represented in this picture. Consequently, it is not clear for the users what is the value of the function at the end points of each interval, for instance, at $x = 2$. Since no graphical hint for this value is provided, the user can be confused about the value of the function at that point. This is actually what usually happens to many of our students, who tend to confuse the meaning of functions such as `Floor`, `Ceiling` and `Round`. For example, think about the point $x = 1.999$. It is clear from Figure 2 that `Round[1.999]=2`. But, what is the

value of the `Floor` function at that point? And the value of the `Ceiling` function at the same point?

```
Plot[#, {x, -3, 3}] & /@ {Floor[x], Ceiling[x], Round[x]}
```

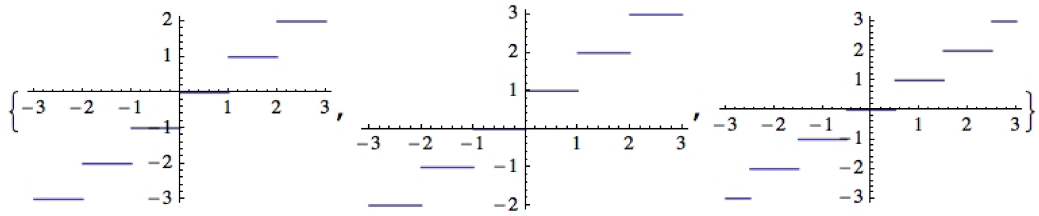

Figure 2: This graphical representation of the `Floor`, `Ceiling` and `Round` functions makes it difficult to determine their values at end points of each interval.

This confusion happens even although the internal representation of the functions is correct. For instance:

```
Assuming[{-3 ≤ x ≤ 3}, PiecewiseExpand[Floor[x]]]
```

$$\begin{cases} -3 & x < -2 \\ -2 & -2 \le x < -1 \\ -1 & -1 \le x < 0 \\ 1 & 1 \le x < 2 \\ 2 & 2 \le x < 3 \\ 3 & x \ge 3 \\ 0 & \text{True} \end{cases}$$

Figure 3: Internal piecewise representation of the `Floor` function on the interval $[-3, 3]$.

To fix this (graphical) problem, we have created two new commands, `PwExpand` and `PwPlot`, that provide a careful analysis of the end points of each interval. To see how do they work, let us take a look at next example of the function $f(x)$ in Fig. 4:

```
f[x_] := { x + 1   -1 < x < 2
         { -1 / 2    x == 2
         { 1 - x   2 < x < 3
         { Null      True
```

Figure 4: Example of a piecewise function.

Mathematica returns this graphical representation for $f(x)$ (see Figure 5):

```
Plot[f[x], {x, -1, 3}]
```

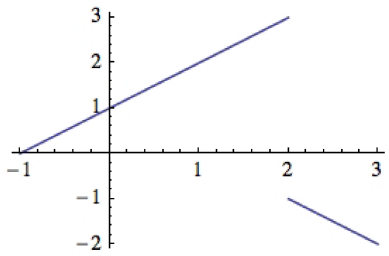

Figure 5: *Mathematica* graphical representation of function $f(x)$ from Figure 4. Compare with the output in Figure 6.

Note that no information about the current value of $f(x)$ at $x = 2$ can be extracted from this picture. Our command `PwPlot` fixes this problem:

```
PwPlot[f[x], {x, -2, 3}]
```

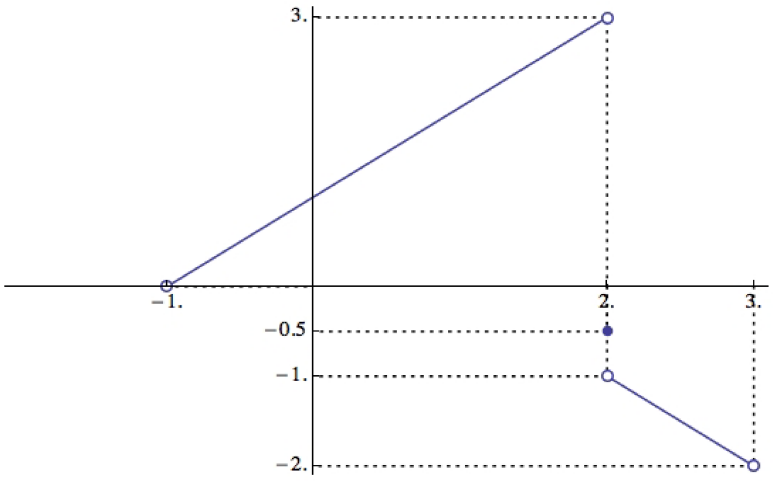

Figure 6: Graphical representation of function $f(x)$ from Figure 4 by using our package.

From this picture, it is clear that the value of the function at $x = 2$ is -0.5. Note also that from the figure it becomes clear that the function is not only discontinuous at that point, but also discontinuous at both sides of the point.

Some other illustrative examples follow. In Fig. 7 we consider a function that involves the `Floor` function. Our command `PwExpand` expands this function as a piece-

wise function. Then, our command `PwPlot` plots the function on its domain. Note the continuity at the endpoint $x = 0$ and the jump discontinuity at $x = 2$. Note also that the function is discontinuous on the left at $x = 2$ but is continuous on the right.

```
f[x_] := { x/2 |x| + 1/2    -3 ≤ x < 2
         { x - |x/2|        2 ≤ x ≤ 6
         { Null             True
```

```
PwExpand[f[x]]
```

$$\begin{cases} \frac{1}{2}\left(1 - x^2\right) & -3 \le x < 0 \\ \frac{1}{2}\left(1 + x^2\right) & 0 \le x < 2 \\ \frac{x}{2} & 2 \le x \le 6 \\ \text{Null} & \text{True} \end{cases}$$

```
PwPlot[f[x], {x, -3, 6}]
```

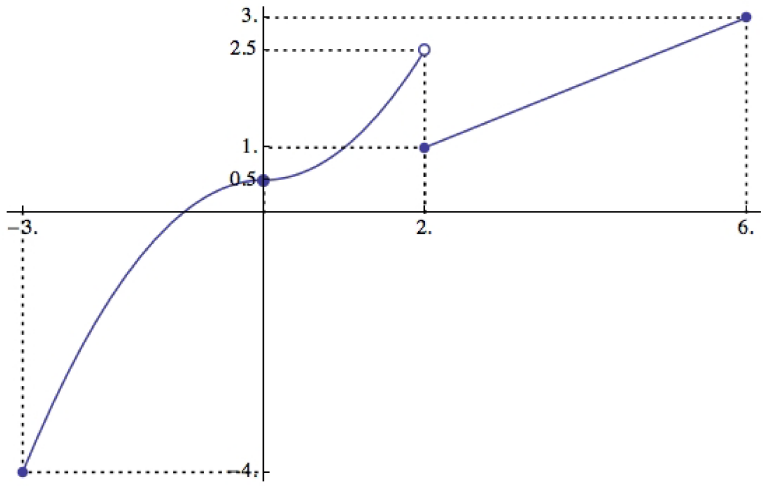

Figure 7: Illustrative example of our *Mathematica* package for piecewise functions.

Last example shows a more complicated function involving the `Floor`, `Sign`, and rational functions. Figure 8 shows its piecewise functional structure and its graphical representation. The reader will notice that now it is much easier to visually identify the different pieces the function is comprised of. Also, the figure helps the reader to figure out what happens at the endpoints of each interval of the function domain.

$$f[x\_] := \begin{cases} x\left|\frac{x^2-1}{2x}\right| & -3 \le x < 2 \\ \mathrm{sgn}\left[\frac{2}{x^2-16}\right] & 2 \le x \le 6 \\ \mathrm{Null} & \mathrm{True} \end{cases}$$

```
PwExpand[f[x]]
```

$$\begin{cases} -1 & 2 \le x < 4 \\ 1 & 4 < x \le 6 \\ \frac{1}{2}\left(1 - x^2\right) & -3 \le x < -1 \;||\; 0 < x < 1 \\ \frac{1}{2}\left(-1 + x^2\right) & -1 \le x < 0 \;||\; 1 \le x < 2 \\ \mathrm{Null} & \mathrm{True} \end{cases}$$

```
PwPlot[f[x], {x, -3, 6}]
```

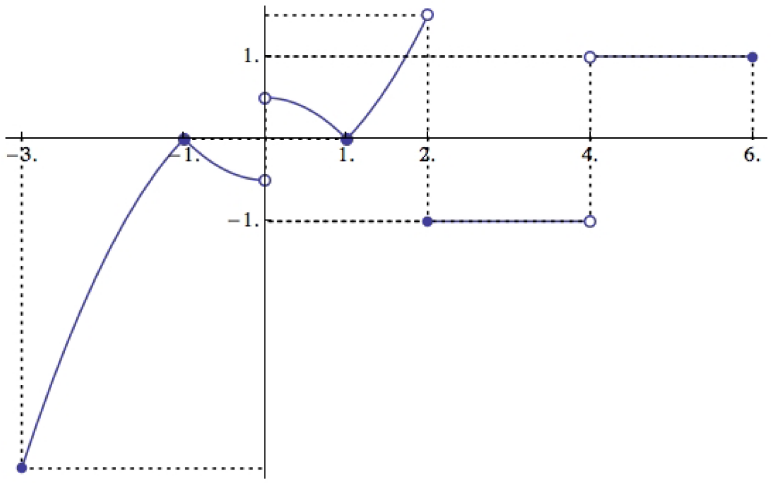

Figure 8: Illustrative example of our *Mathematica* package for piecewise functions.

3.1.2 Algebraic manipulation

Another exciting feature of our package for piecewise functions is the possibility to perform algebraic operations directly with piecewise functions. Let us analyze this issue with an example. Figure 9 shows three piecewise functions labelled as $f(x)$, $g(x)$, and $h(x)$, respectively. We can compute algebraic operations ranging from very simple operations such as $f+g$, to more complicated ones, such as f/g. Even the composition of functions $f \circ g$ can be obtained. Similarly, we can compute combinations of two or more piecewise functions, such as $f * g - h$. Note that the corresponding output is displayed as a piecewise function as well.

```
f[x_] := { 7 x - 3   -3 < x < 0
         { 2 x + 8   0 ≤ x < 5
         { Null      True
```

```
g[x_] := { 2 x - 6   -2 < x < 2
         { x^2 - 2   2 ≤ x < 4
         { Null      True
```

```
h[x_] := { x + 6   -5 < x < 1
         { x - 2   1 ≤ x < 6
         { Null    True
```

```
(f + g)[x]
```

$$\begin{cases} -9 + 9x & -2 < x < 0 \\ 2 + 4x & 0 \le x < 2 \\ 6 + 2x + x^2 & 2 \le x < 4 \\ \text{Null} & \text{True} \end{cases}$$

```
(f / g)[x]
```

$$\begin{cases} \frac{3-7x}{6-2x} & -2 < x < 0 \\ \frac{4+x}{-3+x} & 0 \le x < 2 \\ \frac{2(4+x)}{-2+x^2} & 2 \le x < 4 \\ \text{Null} & \text{True} \end{cases}$$

```
(f ∘ g)[x]
```

$$\begin{cases} -45 + 14x & \frac{3}{2} < x < 2 \\ 4 + 2x^2 & 2 \le x < \sqrt{7} \\ \text{Null} & \text{True} \end{cases}$$

```
(f * g - h)[x]
```

$$\begin{cases} 12 - 49x + 14x^2 & -2 < x < 0 \\ -54 + 3x + 4x^2 & 0 \le x < 1 \\ -46 + 3x + 4x^2 & 1 \le x < 2 \\ -14 - 5x + 8x^2 + 2x^3 & 2 \le x < 4 \\ \text{Null} & \text{True} \end{cases}$$

Figure 9: Some examples of operations with piecewise functions by using our package.

3.2 Matrix Manipulations

Last examples concern the problem of manipulation of matrices by rows or columns. This is a typical example of vector operation that is usually required in many mathe-

matical courses and that students judge as being boring and prone to errors. Debugging is usually performed by hand, and it is therefore tedious and time-consuming. Besides, it could also lead to additional errors because of boredom and tiredness. These problems can readily be fixed by using our package for matrix manipulation.

3.2.1 Row manipulations

This example shows how to operate by rows. Our command `RowOperation` allows us to carry out a number of manipulations by rows. In this case, we replace the first row of the given matrix by the sum of the first two rows:

```
RowOperation[{{0, 1, 1, 1, 0, 0}, {1, 0, 1, 0, 1, 0}, {1, 1, 0, 0, 0, 1}}, R1 + R2 → R1]
```

$$\begin{pmatrix} 1 & 1 & 2 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Figure 10: Example of row manipulation and replacement on a matrix.

Several replacements can be computed simultaneously, even on the matrix obtained in previous execution (indlcated by the symbol % in this input):

```
RowOperation[%, {-R1 + R2 → R2, -R1 + R3 → R3}]
```

$$\begin{pmatrix} 1 & 1 & 2 & 1 & 1 & 0 \\ 0 & -1 & -1 & -1 & 0 & 0 \\ 0 & 0 & -2 & -1 & -1 & 1 \end{pmatrix}$$

```
RowOperation[%, -R2 → R2]
```

$$\begin{pmatrix} 1 & 1 & 2 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & -2 & -1 & -1 & 1 \end{pmatrix}$$

```
RowOperation[%, -R2 + R1 → R1]
```

$$\begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & -2 & -1 & -1 & 1 \end{pmatrix}$$

Figure 11: Series of row manipulations and replacements on previous output.

Some other examples are shown in Figure 12:

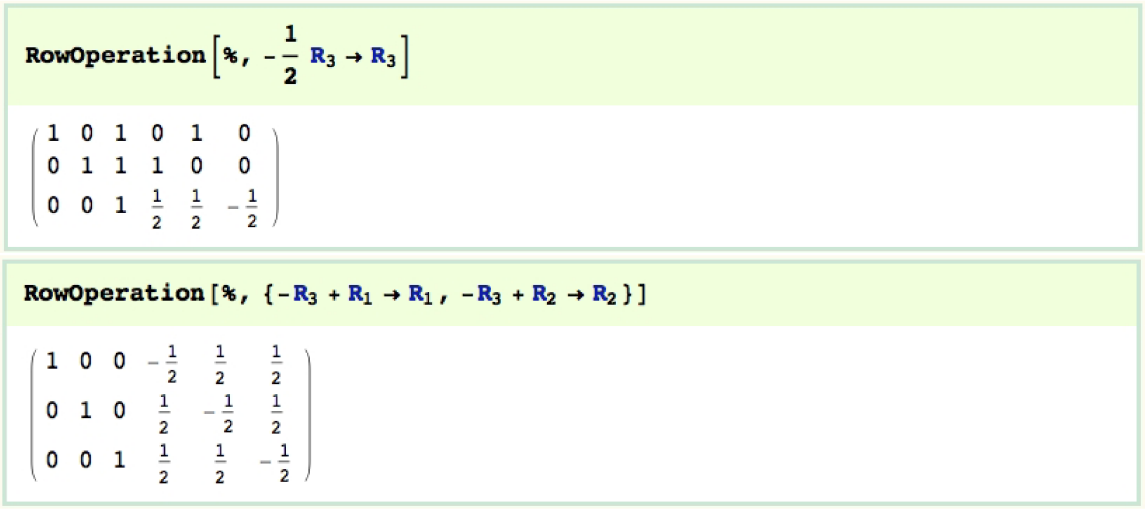

Figure 12: Examples of row manipulations and replacements on previous output.

3.2.2 Nested manipulations

The package also allows nested manipulations into a single command, as shown in next example:

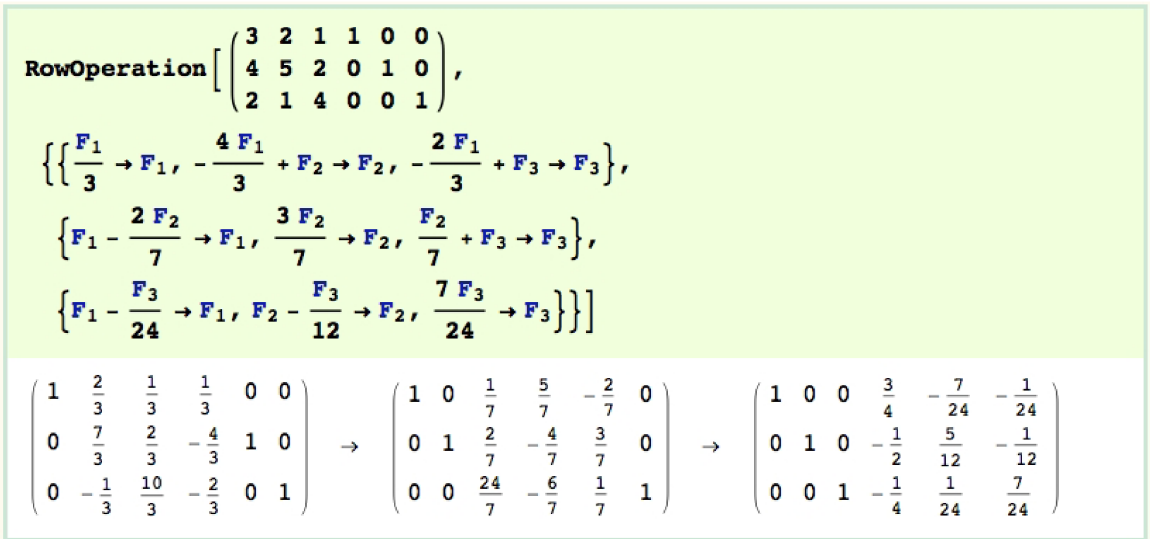

Figure 13: Example of nested manipulations and replacements.

Note the sequence of matrices obtained as the output of the process.

3.2.3 Column manipulations

Of course, manipulations can also be carried out on columns, as shown by this example:

```
ColumnOperation[{{0, 1, 1, 1, 0, 0}, {1, 0, 1, 0, 1, 0}, {1, 1, 0, 0, 0, 1}}, {C1 + C2 → C1, C1 → C2}]
```

$$\begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Figure 14: Example of column manipulations and replacements.

Note the similarity of the input structure for the commands `RowOperation` and `ColumnOperation`, as well as the notation consistency: rows and columns are denoted as $\mathtt{R}_i$ and $\mathtt{C}_j$, respectively.

3.2.4 Column swapping

Swapping of rows and columns is also allowed in our package. Next execution shows an example of column swapping between the first two columns of the previous output matrix:

```
ColumnOperation[%, C1 ↔ C2]
```

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 2 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Figure 15: Example of column swapping.

4 Conclusions and Further Remarks

This paper shows a preview of a new mathematical software developed by the authors, preliminary called *FunMath-MPS* version 0.1. The software is basically an on-going project towards the development of a full suite of Mathematica packages providing support to the computational needs of both teachers and students of the *"Fundamentals of Mathematics"* course, which is traditionally taught in Mathematics, Physics, and other Engineering degrees in many Universities and Colleges all over the world. The talk explored some issues regarding the design of this suite. It also shows some

preliminary examples of the on-going packages in two different subjects: graphical representation and algebraic manipulation of piecewise functions, and row and column operations on matrices.

The presented suite is at a very initial stage, and a lot of work is still needed to accomplish our goals. Future work includes (but is not limited to) the full development of the packages for all topics involved in the course as well the development of a sets of palettes with buttons and other interactive tools for better performance and easier user interaction. Applications of the suite to educational purposes and other fields are also part of our plans for future work.

Acknowledgements

This paper is the printed version of an invited talk delivered by the author at RIMS (Research Institute for Mathematical Sciences) workshop during the *RIMS Workshop on Mathematical Software and Education: Study on Effective Use of Mathematical Software*, Kyoto University (Japan), on August 21st. 2013. The author would like to thank the organizers of this exciting RIMS workshop for their diligent work and kind invitation. Special thanks are owed to Prof. Yasuyuki Nakamura (Nagoya University) for his patience and support regarding the writing of this paper.

This research has been kindly supported by the Computer Science National Program of the Spanish Ministry of Economy and Competitiveness, Project Ref. #TIN2012-30768, Toho University (Funabashi, Japan), and the University of Cantabria (Santander, Spain).

References

[1] Iglesias, A., Gálvez, A.: Effective BD-binding edutainment approach for powering students' engagement at University through videogames and VR technology. In: International Conference on Convergence Information Technology-ICCIT'2008 - Busan (Korea). *IEEE Computer Society Press*, (2008) 307-314.

[2] Iglesias, A., Ipanaqué, R.: Using computer algebra systems to achieve Bologna's Declaration educational goals. A case study: symbolic proof of limits of functions. *International Journal of Computer Science and Software Technology*, **2**(1) (2009) 35-42.

[3] Iglesias, A.: Facing the challenges of the new European Space of Higher Education through effective use of computer algebra systems as an educational tool. *RIMS Kokyuroku Journal Series*, **1624** (2009) 114-128.

[4] Iglesias, A.: Computer technologies for XXI century education: A new way to communicate and learn at the University of Cantabria. *RIMS Kokyuroku Journal Series*, **1674** (2010) 53-67.

[5] Iglesias, A.: Combining functional equations and computer algebra systems with regard to XXI century Mathematics education. *RIMS Kokyuroku Journal Series*, **1735** (2011) 213-223.

[6] Iglesias, A.: Freeware, shareware, and open-source mathematical software tools: a feasible alternative to commercial symbolic packages for Mathematics education. *RIMS Kokyuroku Journal Series*, **1865** (2013) 204-214.