Facing the Challenges of the New European Space of Higher Education Through Effective Use of Computer Algebra Systems as an Educational Tool

Andrés Iglesias
Department of Applied Mathematics and Comp. Sciences
University of Cantabria, Avda. de los Castros
s/n, E-39005, Santander, Spain
iglesias@unican.es
http://personales.unican.es/iglesias

Abstract

Academic institutions of most European countries are now in the midst of the process of overall convergence of studies at university level, the so-called *European Space of Higher Education*. This process, born from Bologna's declaration, implies a dramatic change in our way of teaching and learning. This change is specially challenging in very difficult subjects such as Mathematics and other scientific disciplines, which typically suffer from high failure rates. It's author's opinion that Computer Algebra Systems are very powerful tools (arguably the best ones) to face the challenges of this new approach to Higher Education. To support such claims, some examples of *Mathematica* packages developed by the author during the last few years for teaching different subjects in scientific degrees (according to the principles and regulations of the European Space of Higher Education) are briefly described.

1 Introduction

Since its signature in 1999, Bologna's Declaration has epitomized the much-needed, largely-demanded process of changes undertaken in most European countries in order to reform their Higher Education system. This agreement is not only designed for new curricula and academic regulations; it also accounts for a new approach to teaching and learning (in author's opinion, the most important issue of this process). 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.

Right or not, the truth is Mathematics is very much practice-based. Students may get 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. This paper aims at supporting such claims by presenting some examples of *Mathematica* packages developed by the author during the last few years for teaching different subjects in scientific degrees according to the principles and regulations of the European Space of Higher Education.

2 Some Illustrative Examples

2.1 Example 1: Binary Distillation Column

This section is about a *Mathematica* program for distillation column design described originally in [7]. This example is intended for students of Chemical Engineering degree who can use a CAS as a useful tool for their symbolic/numerical computations and graphical output. For the sake of simplicity, we consider the case of continuous distillation columns for binary mixtures. However, our approach is very general as it can be applied to any other kind of distillation columns by simply replacing our assumptions by those of each specific case. In this example, we assume that the columns are designed through McCabe-Thiele's procedure [11, 12]. This is a graphical method which determines the number of stages required for the desired degree of separation and the location of the feed tray as functions of some parameters of the problem. The program is general enough to analyze a number of different mixtures under different conditions as well as the role of many relevant parameters of this process. To this purpose, an adequate combination of symbolic and numerical calculations is achieved. From these calculations, both numerical and graphical outputs are obtained. In fact, the graphical output is actually a *Mathematica* movie of McCabe-Thiele's diagram.

The method has been implemented as a *Mathematica* package, called `MTBDC.m`, where this acronym stands for *M*cCabe-*T*hiele *B*inary *D*istillation *C*olumn. This section describes one example of application of this program to a equimolar (i.e. $x_f = 0.5$) binary mixture whose relative volatility is $\alpha = 4$. Our target is to design a distillation column that obtains a destillate with 85 % of purity (i.e. $x_d = 0.85$) and bottoms with 5 % of purity (i.e. $x_b = 0.05$). The first comand `Reflux[xd,xf,alpha]` determines three values: the reflux, and the liquid and vapor compositions of the more volatile component:

```
In[1]:= Needs["MCBDC‘"]
In[2]:= Reflux[0.85,0.5,4]
Out[2] := {1.75, 0.2, 0.5}
```

Then, the command `OperatingLines[xd,xf,alpha]` is applied to the calculation of the operating lines of the rectification section and the stripping section, shown in Fig. 1(d). The third step is to apply McCabe-Thiele's method to determine the number of

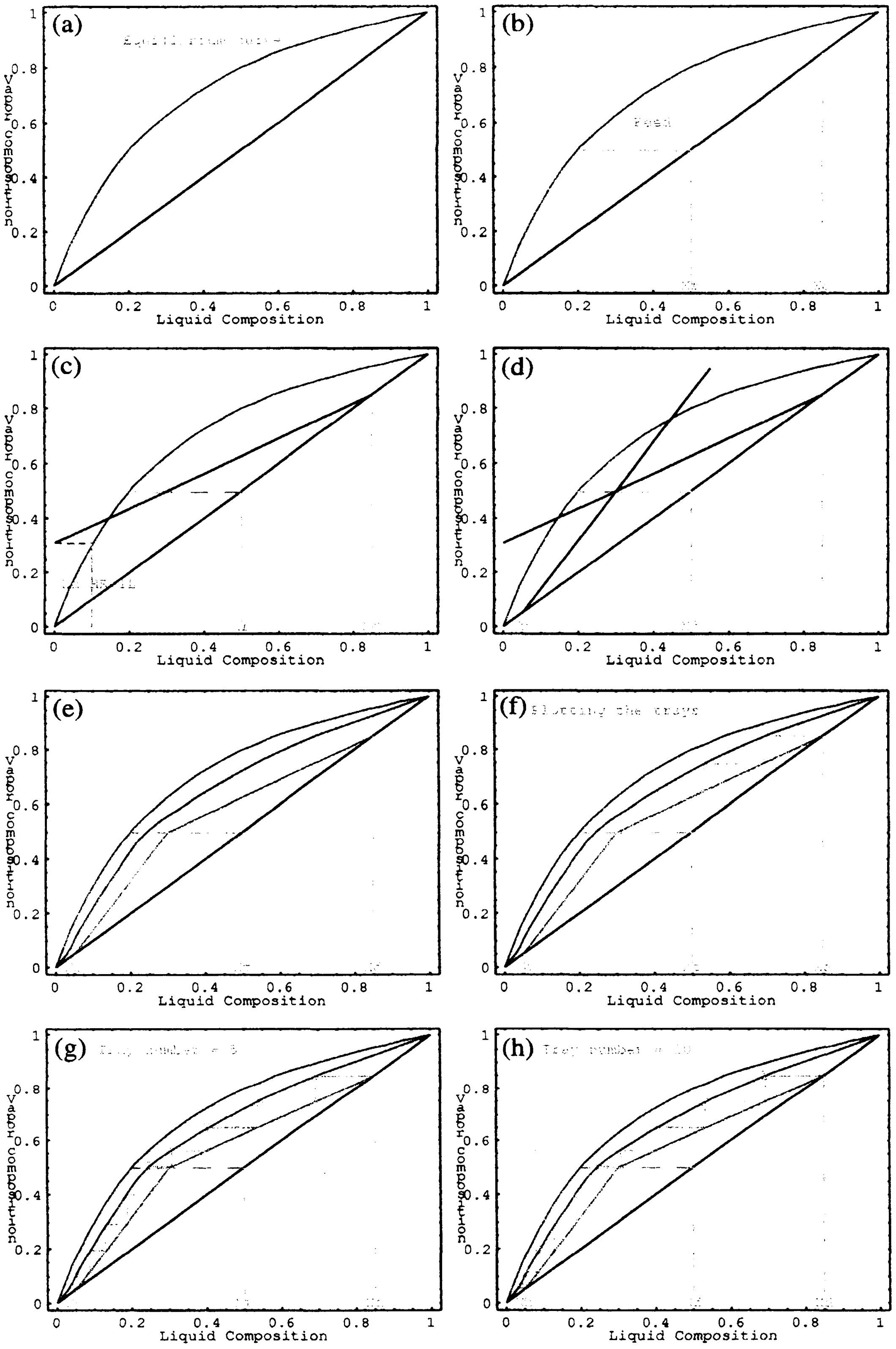

Figure 1: McCabe-Thiele's method for binary distillation column design: (a)-(h) different steps of the method (see the body text for details).

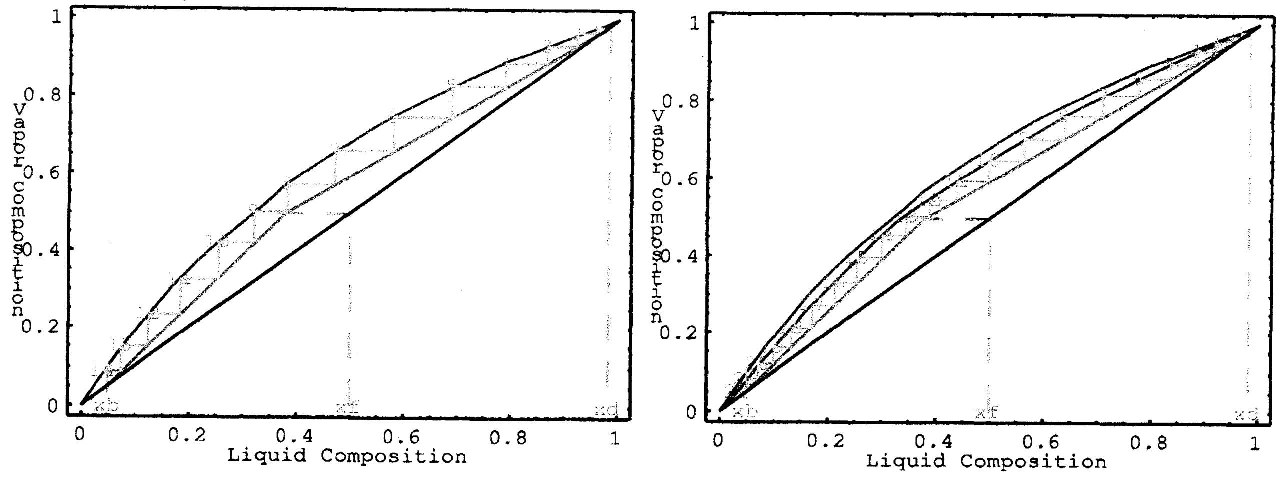

Figure 2: McCabe-Thiele's method for an equimolar heptane-octane mixture with $x_d = 0.98$, $x_f = 0.5$, $x_b = 0.05$ and $\alpha = 2.17$: (left) $E_f = 1$; (right) $E_f = 0.65$.

trays of the column as well as the location of the feed tray. To this end, we define the `MTPlot` command, which accepts five different arguments, namely, x_d, x_f, x_b, α and the efficiency E_f. For example, `MTPlot[0.85,0.5,0.05,4,0.6]` returns a sequence of 49 frames corresponding to the different steps of McCabe-Thiele's method. Eight of these frames are displayed in Figure 1.

One of the most remarkable *Mathematica* features in this work is the possibility to generate QuickTime^{TM} movies. For example, the different images in Figure 1 correspond to eight frames of a QuickTime movie that reproduces McCabe-Thiele's method in a graphical way. The movie is automatically generated from the `MTPlot` command. Different values of its arguments are associated with different initial mixtures and/or final products. This option is specially valuable for educational and training purposes, as we can obtain a virtually unlimited set of distillation columns in a fast and simple way.

McCabe-Thiele's method does not allow us to determine all relevant distillation column parameters and thus, some additional equations must also be considered. The discussion is beyond the scope of this paper and will not be included here. At this time it is enough to say that a number of different *Mathematica* commands incorporating these additional equations have been implemented in the `MTBDC.m` package. For example, if we start with an equimolar mixture of heptane-octane with a relative volatility $\alpha = 2.17$ and efficiency $E_f = 1$ and we wish to obtain a destillate with 98 % of heptane and bottoms with only a 5% of heptane, the column parameters can be determined as:

```
In[3]:= ColumnParameters[0.98,0.05,0.5,2.17,1]
Out[3]:= Number of trays = 14 (rebolier included)
         Number of Feed tray = 9
         Column height = 8.4 meters
         Distance between trays = 0.6 meters
```

Figure 2 shows the McCabe-Thiele's diagram of the column for total efficiency $E_f = 1$ (left) and when the efficiency of the process is only a 65 %, that is $E_f = 0.65$ (right). In the first case, the column has only 14 trays and its height is 8.4 meters, while in the second case the number of trays increases up to 22 and the column height is now 13.2 meters.

2.2 Example 2: Solving Inequalities

Solving inequalities is a very important topic in Mathematics, with outstanding applications in many areas of theoretical and applied science. Inequalities play a key role because many problems cannot be completely and accurately described by using only equalities. However, since there is not a general methodolody for solving inequalities, their symbolic computation is still a challenging problem in computational algebra. Depending on the kind of functions involved, there are many "specialized" methods such as cylindrical algebraic decomposition, Gröebner basis, quantifier elimination, etc. [1, 6, 8, 13].

This section uses a nonstandard *Mathematica* package [9], `InequationPlot`, for displaying the two-dimensional solution sets of several inequalities. In particular, it extends *Mathematica*'s capabilities by providing graphical solutions to many inequalities (such as those involving trigonometric, exponential and logarithmic functions) that cannot be solved by using the standard *Mathematica* commands and packages [14, 15]. The package also deals with inequalities involving complex variables by displaying the corresponding solutions on the complex plane.

Inequalities involving trigonometric functions cannot be solved by standard *Mathematica* commands. For example, let us try to display the solution sets of each of the inequalities

$$sin(x+y) > \frac{1}{2} \tag{1}$$

and

$$sin(2x) + cos(3y) < 1 \tag{2}$$

on the set $[-8,8] \times [-8,8]$ by using the standard *Mathematica* commands. In this case, we must use the command `InequalityPlot` of the *Mathematica* package:

```
In[4]:=<<Graphics'InequalityGraphics'
```

Unfortunately, since the region defined by inequality (1) on the prescribed domain cannot be broken down into cylinders, *Mathematica* fails to give the solution:

```
In[5]:=InequalityPlot[Sin[x + y] >1/2, {x,-8,8},{y,-8,8}]
```

Out[5] := *InequalityPlot* :: *region* :
The region defined by $sin(x+y) > 1/2 \wedge -8 <= x <= 8 \wedge -8 <= y <= 8$ *could not be broken down into cylinders.*

On the contrary, the previous inequalities can be solved by loading our package:

```
In[6]:=<<InequationPlot'
```

which includes the command

`InequationPlot[`*ineqs*, {*x*, *xmin*, *xmax*}, {*y*, *ymin*, *ymax*}, *opts*`]`

for displaying the two-dimensional region of the set of points satisfying the inequalities *ineqs* of real numbers inside the square $[xmin, xmax] \times [ymin, ymax]$. For example, inequalities (1)-(2) can be solved as follows:

```
In[7]:=InequationPlot[#,{x,-8,8},{y,-8,8},AspectRatio->Automatic]& /@
        {Sin[x+y]>1/2,Sin[2 x]+Cos[3 y]<1}
```

Out[7]:= See Figure 3

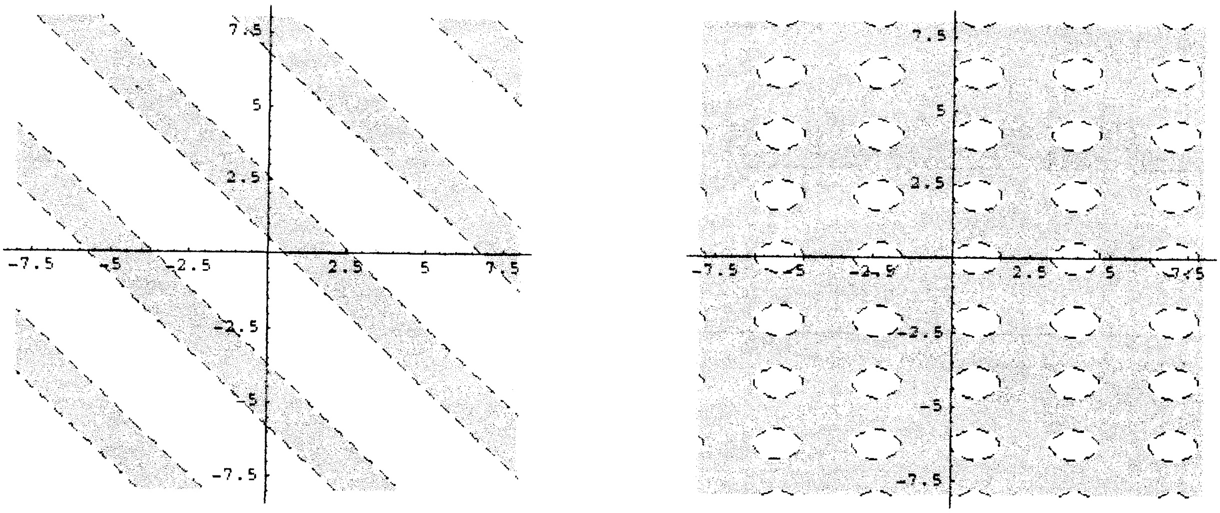

Figure 3: Some examples of inequality solutions on the square $[-8,8] \times [-8,8]$: (left) $sin(x+y) > \frac{1}{2}$; (right) $sin(2x) + cos(3y) < 1$.

Similarly, Fig. 4 displays the solution sets of the inequalities $F(x) + F(y) = 1$ and $F(x^2) + F(y^2) = 1$ (where F stands for the floor function) on the squares $[-4,4] \times [-4,4]$ and $[-2,2] \times [-2,2]$, respectively. We would like to remark that the *Mathematica* command `InequalityPlot` does not provide any solution for these inequalities either.

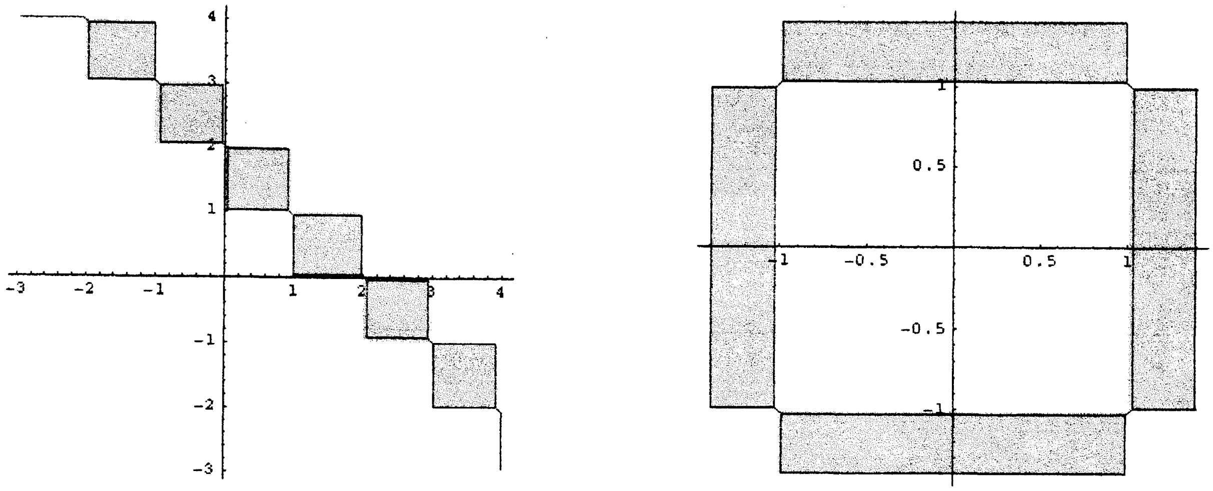

Figure 4: Some examples of inequality solutions: (left) $floor(x) + floor(y) = 1$ on the square $[-4,4] \times [-4,4]$; (right) $floor(x^2) + floor(y^2) = 1$ on the square $[-2,2] \times [-2,2]$.

The previous command, `InequationPlot`, can be generalized to inequalities involving complex numbers. The new command

`ComplexInequationPlot`$[ineqs, \{z, \{Rezmin, Rezmax\}, \{Imzmin, Imzmax\}\}, opts]$

displays the solution sets of the inequalities *ineqs* of complex numbers inside the square in the complex plane given by $[Rezmin, Rezmax] \times [Imzmin, Imzmax]$. In this case, the functions appearing within the inequalities need to be real-valued functions of a complex argument, e.g. *Abs*, *Re* and *Im*. For example:

```
In[8]:=ComplexInequationPlot[#,{z,{-2,3},{-3,3}},AspectRatio->Automatic]&
   /@ {1<Abs[z^2-z+1]<4,1<Abs[z^2-2z]/Abs[z^2+3]<4}
```

Out[8]:= See Figure 5

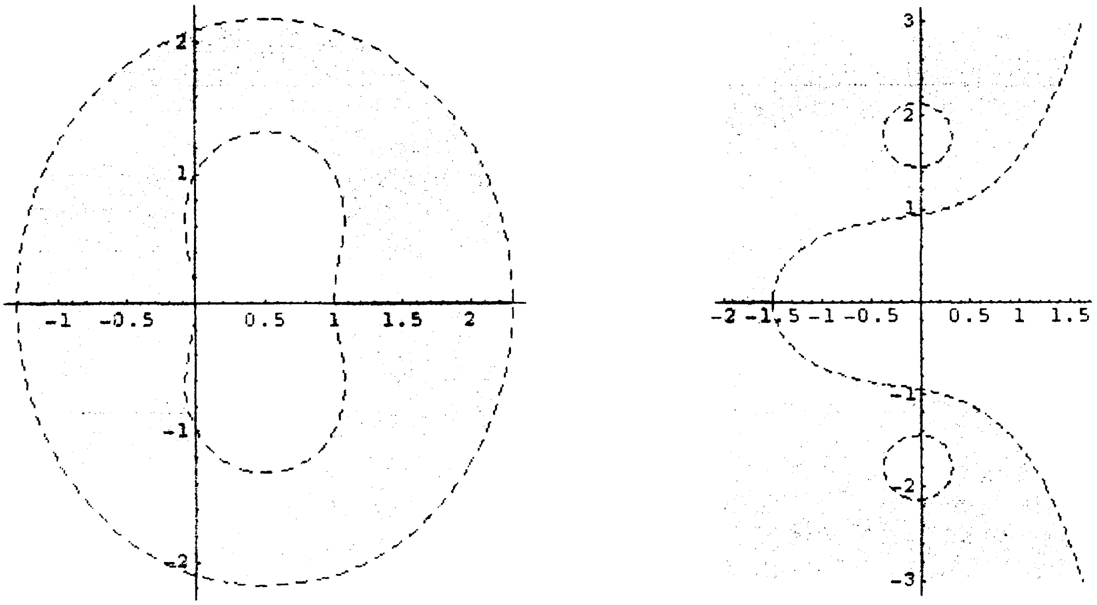

Figure 5: Some examples of inequality solutions for $z = a + bi \in \mathbb{C}$ such that $a \in [-2, 3]$ and $b \in [-3, 3]$: (left) $1 < ||z^2 - z + 1|| < 4$; (right) $1 < \dfrac{||z^2 - 2z||}{||z^2 + 3||} < 4$.

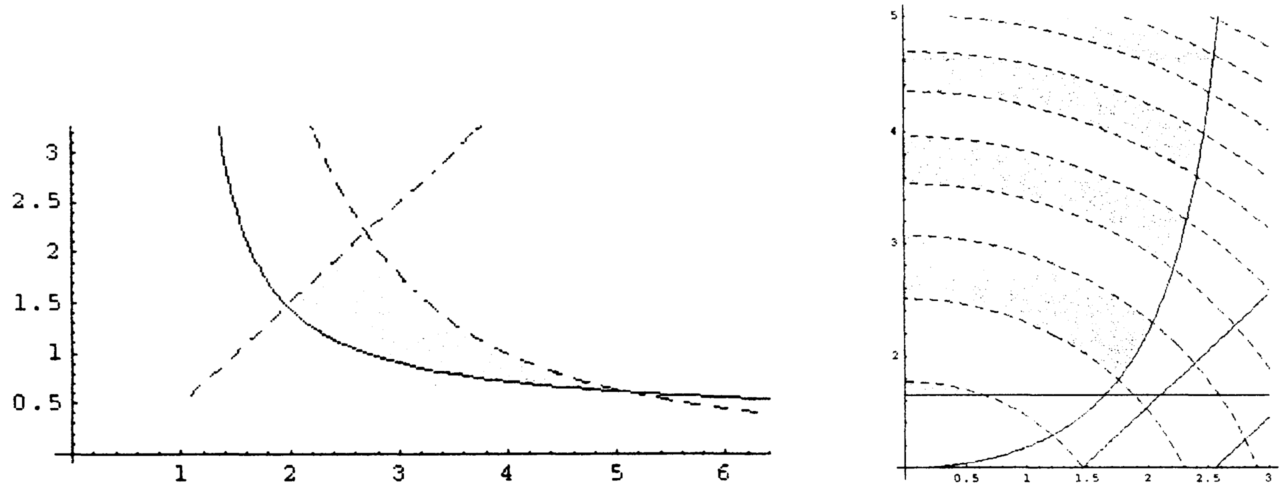

Figure 6: Solution sets for the inequality systems: (left) Eq. (3); (right) Eq. (4).

The last example shows how complicated the inequality systems can be: in addition to include exponential, logarithmic and trigonometric functions, combinations and even compositions of these (and other) functions can also be considered. In Figure 6 the solutions sets of the inequality systems:

$$e^y \geq 1 \bigwedge log(x)y \geq 1 \bigwedge x\sqrt{y} < 4 \bigwedge x - y > \frac{1}{2} \tag{3}$$

$$log(y) \geq \frac{1}{2} \bigwedge sin(x)y \geq x \bigwedge cos(e^{x-y}) \geq 0 \bigwedge sin(x^2 + y^2) > 0 \tag{4}$$

on $[1, 10] \times [0, 10]$ and $[0, 3] \times [1, 5]$ respectively are displayed.

2.3 Example 3: Gauss Map of Surfaces

This section focuses on a classical topic in the field of Differential Geometry: the visualization of the *Gauss map* of a surface. Roughly speaking, the Gauss map projects surface normals onto a unit sphere, providing a powerful visualization of the geometry of a graphical object. On the other hand, dynamic visualization of the Gauss map speeds understanding of complex surface properties. This section applies a *Mathematica* package described in [10], GaussMap, for computing and displaying the tangent and normal vector fields and the Gauss map of surfaces described symbolically in either implicit or parametric form. Firstly, we load the package:

```
In[9]:=<<DifferentialGeometry'GaussMap'
```

2.3.1 Implicit Surfaces

The ImplicitNormalField command calculates the normal vector field of any implicit surface in the form $f(x, y, z) = 0$ and returns a graphical output comprised of the surface and such a normal vector field. The first example is given by the paraboloid $x^2+y^2-z = 0$:

```
In[10]:=ImplicitNormalField[x^2+y^2-z,{x,-2,2},{y,-2,2},{z,-1,2},
    Surface->True,PlotPointsSurface->{4,7},VectorHead->Polygon]
```

Out[10] := *See Figure* 7

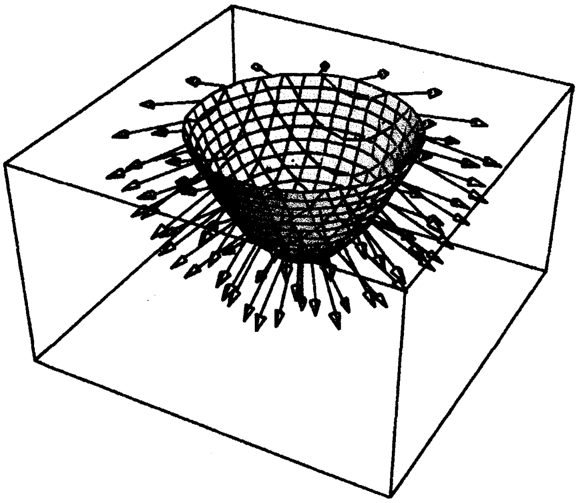

Figure 7: The paraboloid $x^2 + y^2 - z = 0$ and its vector normal field.

The `ImplicitGaussMap` command of an implicit surface returns the original surface and its Gauss map on the unit sphere:

```
In[11]:=ImplicitGaussMap[x^2+y^2-z,{x,-2,2},{y,-2,2},{z,-1,2},
        PlotPoints->{4,7}]
```

Out[11] := *See Figure* 8

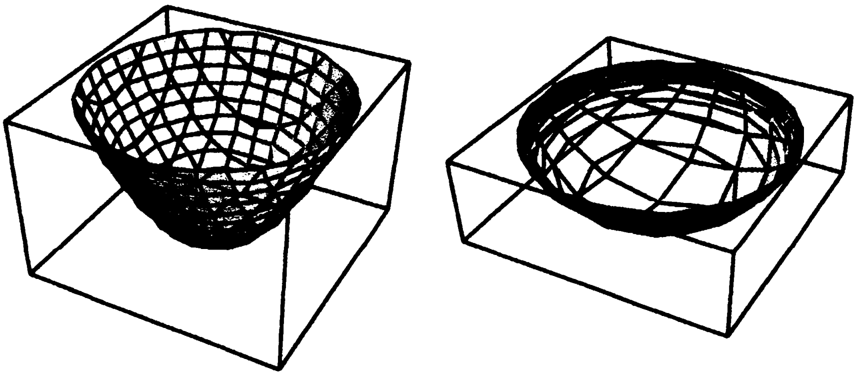

Figure 8: (left) Implicit surface $x^2 + y^2 - z = 0$; (right) its Gauss map.

Next example calculates the normal vector field of the surface $z - x^2 + y^2 = 0$:

```
In[12]:=ImplicitNormalField[z-x^2+y^2,{x,-2,2},{y,-2,2},{z,-2,2},
     Surface->True, PlotPointsSurface->{4,6},VectorHead->Polygon,
     PlotPointsNormalField->{3,4}]
```

Out[12] := *See Figure* 9

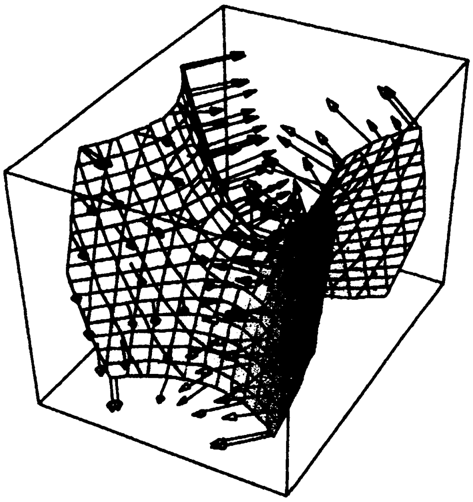

Figure 9: The surface $z - x^2 + y^2 = 0$ and its vector normal field.

The Gauss map image of such a surface is obtained as:

```
In[13]:=ImplicitGaussMap[z-x^2+y^2,{x,-2,2},{y,-2,2},{z,-2,2},
        PlotPoints->{4,6}]
```

Out[13] := *See Figure* 10

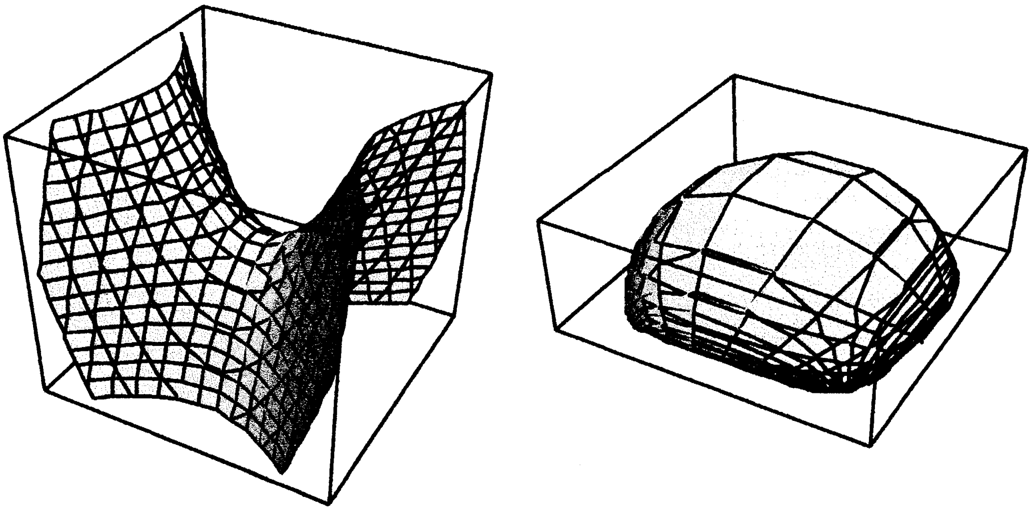

Figure 10: (left) Implicit surface $z - x^2 + y^2 = 0$; (right) its Gauss map.

A more complicated example is given by the double torus surface, defined implicitly by $\left(\sqrt{\left(\sqrt{x^2+y^2}-2\right)^2+y^2}-2\right)^2+z^2-1=0$ and whose normal vector field is shown in Figure 11:

```
In[14]:=ImplicitNormalField[(Sqrt[(Sqrt[x^2+y^2]-2)^2+y^2]-2)^2+z^2-1,
       {x,-6,6},{y,-4,4},{z,-1,1},Surface->True,VectorHead->Polygon,
       PlotPointsSurface->{5,5}]
```

Out[14] := *See Figure* 11

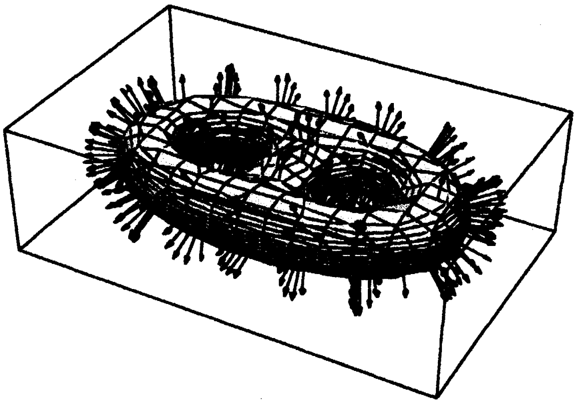

Figure 11: The double torus surface and its vector normal field.

```
In[15]:=ImplicitGaussMap[(Sqrt[(Sqrt[x^2+y^2]-2)^2+y^2]-2)^2+z^2-1,
       {x,-6,6},{y,-4,4},{z,-1,1},PlotPoints->{5,5}]
```

Out[15] := *See Figure* 12

It is interesting to remark that, because the Gauss map is a continuous (in fact, differentiable) function, it is closed for compact sets, i.e. it transforms compact sets into

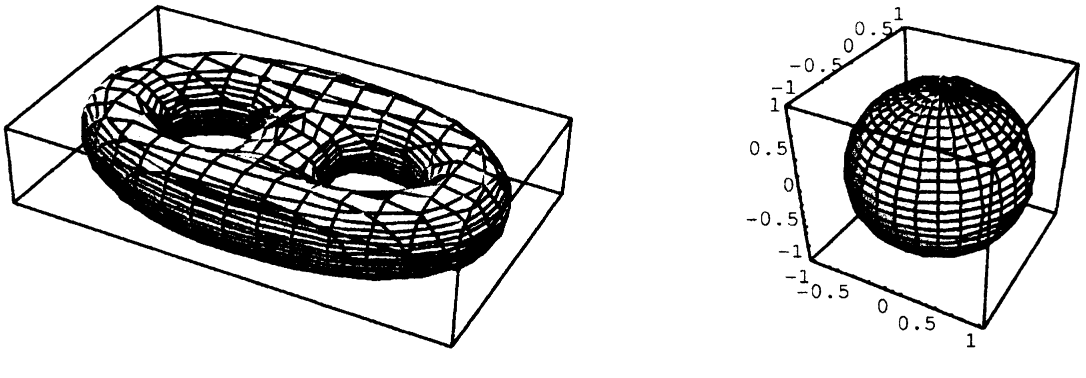

Figure 12: (left) Double torus surface; (right) its Gauss map.

compact sets. Since the double torus is compact, its image is actually the unit sphere. Another example of this property is the Lamé surface of fourth degree $x^4 + y^4 + z^4 = 1$:

```
In[16]:=ImplicitNormalField[x^4+y^4+z^4-1,{x,-1,1},{y,-1,1},{z,-1,1},
     Surface->True, PlotPointsSurface->{4,6},VectorHead->Polygon]
```

Out[16] := *See Figure* 13

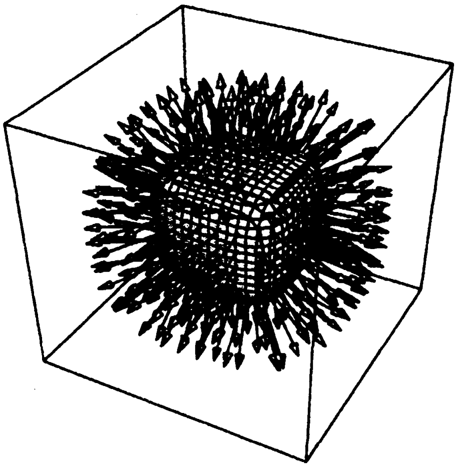

Figure 13: The surface $x^4 + y^4 + z^4 = 1$ and its vector normal field.

Its corresponding Gauss map image can displayed as:

```
In[17]:=ImplicitGaussMap[x^4+y^4+z^4-1,{x,-1,1},{y,-1,1},
        {z,-1,1},PlotPoints->{4,6}]
```

Out[17] := *See Figure* 14

2.3.2 Parametric Surfaces

As indicated above, the package also deals with surfaces given in parametric form. In the following example, we consider the Möbius strip, parameterized by

$$\vec{S}(u,v) = \left(cos(u) + v\,cos(u)sin\left(\frac{u}{2}\right), sin(u) + v\,sin\left(\frac{u}{2}\right)sin(u), vcos\left(\frac{u}{2}\right)\right)$$

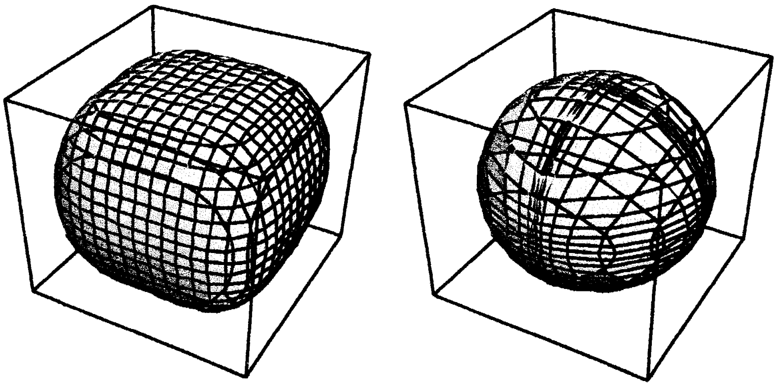

Figure 14: Lamé surface of fourth degree; (right) its Gauss map.

Figures 15 and 16 show its normal vector field and the Gauss map, respectively.

```
In[18]:= ParametricNormalField[{Cos[u]+v Cos[u] Sin[u/2],
     Sin[u]+v Sin[u/2] Sin[u],v Cos[u/2]},{u,0,4 Pi,Pi/10},
     {v,-1/2,1/2,0.1},Surface->True,PlotPoints->{50,7},
     SurfaceDomain->{{0,2 Pi},{-1/2,1/2}},VectorHead->Polygon,
     HeadLength->0.25,HeadWidth->0.1,VectorColor->RGBColor[1,0,0]]
```

Out[18] := *See Figure* 15

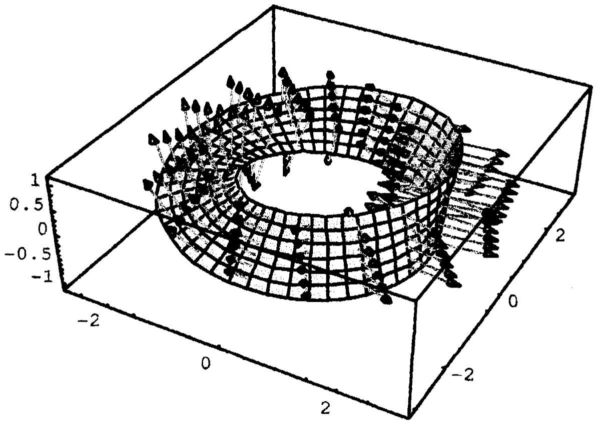

Figure 15: The Möbius strip and its vector normal field.

```
In[19]:=ParametricGaussMap[{Cos[u]+v Cos[u] Sin[u/2],Sin[u]+
     v Sin[u/2] Sin[u],v Cos[u/2]},{u,0,4 Pi},{v,-1/2,1/2},
     SurfaceDomain->{{0,2 Pi},{-1/2,1/2}},PlotPoints->{50,7}]
```

Out[19] := *See Figure* 16

2.4 Example 4: Solving Functional Equations

In this section we use the *Mathematica* package FSolve [3, 4], implemented to solve functional equations [2, 5]. Let's start loading the package:

```
In[20]:=<<FunctionalEquations`FSolve`
```

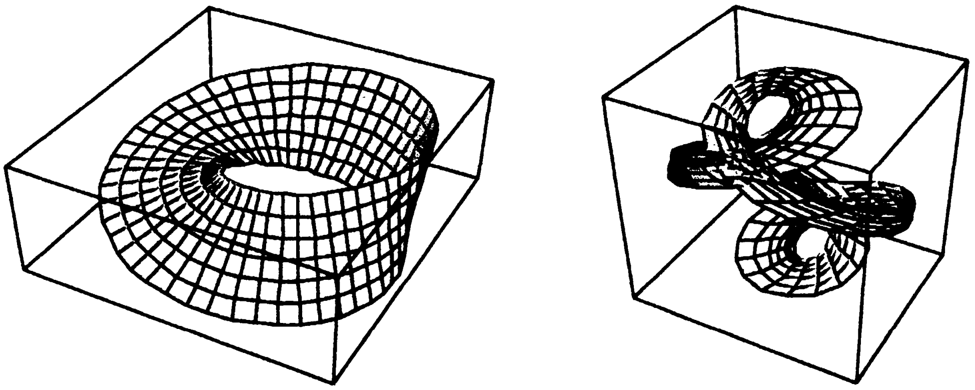

Figure 16: (left) Möbius strip surface; (right) its Gauss map.

It includes the command: `FSolve[`*eqn*, $\{functions\}$, $\{variables\}$, *options*`]` where *eqn* denotes the functional equation to be solved, $\{functions\}$ is the list of unknown functions, $\{variables\}$ is the list of variables and *options* allows the users to consider different domains for the variables (see Table 1) and classes of feasible functions (see Table 2).

Domains	*Package names*	*Definition*
$\mathbb{R}$	`Arbitrary`	$\mathbb{R}$
$\mathbb{R}$	`Real`	$\mathbb{R}$
$\mathbb{I}$	`Irrational`	$\mathbb{R}-\mathbb{Q}$
$\mathbb{Q}$	`Rational`	$\mathbb{Q}$
$\mathbb{Z}$	`Integer`	$\mathbb{Z}$
$\mathbb{R}-\{0\}$	`NonZeroReal`	$\mathbb{R}-\{0\}$
$\mathbb{Q}-\{0\}$	`NonZeroRational`	$\mathbb{Q}-\{0\}$
$\mathbb{Z}-\{0\}$	`NonZeroInteger`	$\mathbb{Z}-\{0\}$
$\mathbb{R}_+$	`RealPositiveZero`	$\{x\in\mathbb{R}/x\geq 0\}$
$\mathbb{Q}_+$	`RationalPositiveZero`	$\{x\in\mathbb{Q}/x\geq 0\}$
$\mathbb{Z}_+$	`IntegerPositiveZero`	$\{x\in\mathbb{Z}/x\geq 0\}$
$\mathbb{R}_{++}$	`RealPositive`	$\{x\in\mathbb{R}/x>0\}$
$\mathbb{I}_{++}$	`IrrationalPositive`	$\{x\in\mathbb{R}-\mathbb{Q}/x>0\}$
$\mathbb{Q}_{++}$	`RationalPositive`	$\{x\in\mathbb{Q}/x>0\}$
$\mathbb{Z}_{++}$	`IntegerPositive`	$\{x\in\mathbb{Z}/x>0\}$
$\mathbb{R}_-$	`RealNegativeZero`	$\{x\in\mathbb{R}/x\leq 0\}$
$\mathbb{Q}_-$	`RationalNegativeZero`	$\{x\in\mathbb{Q}/x\leq 0\}$
$\mathbb{Z}_-$	`IntegerNegativeZero`	$\{x\in\mathbb{Z}/x\leq 0\}$
$\mathbb{R}_{--}$	`RealNegative`	$\{x\in\mathbb{R}/x<0\}$
$\mathbb{I}_{--}$	`IrrationalNegative`	$\{x\in\mathbb{R}-\mathbb{Q}/x<0\}$
$\mathbb{Q}_{--}$	`RationalNegative`	$\{x\in\mathbb{Q}/x<0\}$
$\mathbb{Z}_{--}$	`IntegerNegative`	$\{x\in\mathbb{Z}/x<0\}$

Table 1: List of all feasible domains used in the `FSolve` package.

For instance, we can calculate the solution of the functional equation $f(x+y)=$

Name	*Classes of functions*
`Differentiable`	Differentiable
`Invertible`	Invertible
`Increasing`	Increasing
`Decreasing`	Decreasing
`Continuous`	Continuous
`ContinuousPoint`	Continuous at a point
`Monotonous`	Monotonous
`Arbitrary`	Arbitrary

Table 2: Classes of feasible functions used in the `FSolve` package.

$g(x) + h(y)$ where $x, y \in \mathbb{R}$ and f, g, h are continuous functions as:

```
In[21]:= FSolve[f[x+y]==g[x]+h[y],{f,g,h},{x,y},Domain->Real,
             Class->Continuous]
```

$$Out[21] := \{f(x) \to C(1)\,x + C(2) + C(3), g(x) \to C(1)\,x + C(2), h(x) \to C(1)\,x + C(3)\}$$

where $C(1)$, $C(2)$ and $C(3)$ are arbitrary constants. Note that a single equation can determine several unknown functions (such as f, g and $h$ in this example). Note also that the solution can depend on one or more arbitrary constants and/or arbitrary functions:

```
In[22]:=FSolve[f[x]*Sin[y]+h[x]*g[y]==0,{f,g,h},{x,y}]
```

$$Out[22] := \begin{array}{l} \{\{f[x] \to 0, g[y] \to C[0]Sin[y] + C[1]Arb[0][y], h[x] \to 0\}, \\ \{f[x] \to C[1]Arb[0][x], g[y] \to -\dfrac{C[1]Sin[y]}{C[0]}, h[x] \to C[0]Arb[0][x]\}\} \end{array}$$

```
In[23]:=FSolve[k[u]*l[v]==-b[u],{k,l,b},{u,v}]
```

$$Out[23] := \begin{array}{l} \{\{k[u] \to 0, b[u] \to 0, l[v] \to C[0] + C[1] * Arb[0][v]\}, \\ \{k[u] \to C[0] * Arb[0][u], b[u] \to C[1] * Arb[0][u], l[v] \to -(C[1]/C[0])\}\}, \end{array}$$

```
In[24]:=FSolve[f[g[x]+h[y]]-s[r[x]+s[y]]==0,{f,g,h,r,s},{x,y},
           Domain->RealPositive,Class->Arbitrary]
```

$$Out[24] := \{\{f[x] \to s[\tfrac{x - \mathrm{C}[2] - \mathrm{C}[3]}{\mathrm{C}[1]}], g[x] \to \mathrm{C}[2] + \mathrm{C}[1]\, r[x], h[x] \to \mathrm{C}[3] + \mathrm{C}[1]\, s[x]\}\}$$

Acknowledgments

This paper is the printed version of an invited talk delivered by the author at RIMS (Research Institute for Mathematical Sciences) workshop on *Computer Algebra Systems and Education: A Research about Effective Use of CAS in Mathematics Education*, Kyoto University (Japan), August 29th. 2008. The author would like to thank the organizers of this exciting RIMS workshop for their diligent work and kind invitation. Special thanks are owe to Prof. Setsuo Takato (Toho University, Japan) for his friendship, his great support and hospitality and for making my stay in Kyoto such a wonderful and unforgettable experience. It was indeed a lovely time I will never forget. *Doomo arigatoo gozaimashita! Ookini!*

This research has been supported by the Computer Science National Program of the Spanish Ministry of Education and Science, Project Ref. #TIN2006-13615 and the University of Cantabria.

References

[1] Beckenbach, E.F., Bellman, R.E.: *An Introduction to Inequalities.* Random House, New York (1961).

[2] Castillo, E., Gutiérrez, J.M., Iglesias, A.: Solving a functional equation. *The Mathematica Journal,* **5**(1) (1995) 82-86.

[3] Castillo, E., Iglesias, A., Cobo, A.: A package for symbolic solutions of functional equations. In: Keranen, V., Mitic, P. (Eds.): Proceedings of the First International Mathematica Symposium. *Computational Mechanics Publications*, Southampton (England), pp. 85-92.

[4] Castillo, E., Iglesias, A.: A package for symbolic solution of real functional equations of real variables. *Aequationes Mathematicae,* **54** (1997) 181-198.

[5] Castillo, E., Iglesias, A., Ruiz-Cobo, R.: *Functional Equations in Applied Sciences.* Elsevier Pub., Amsterdam (2005).

[6] Caviness, B.F., Johnson, J.R.: *Quantifier Elimination and Cylindrical Algebraic Decomposition.* Springer-Verlag, New York (1998).

[7] Gálvez, A., Iglesias, A.: Binary distillation column design using Mathematica. *Lectures Notes in Computer Science,* **2657** (2003) 848-857.

[8] Hardy, G.H., Littlewood, J.E., Pólya, G.: *Inequalities* (Second Edition). Cambridge University Press, Cambridge (1952).

[9] Ipanaqué, R., Iglesias, A.: A Mathematica package for solving and displaying inequalities. *Lectures Notes in Computer Science,* **3039** (2004) 303-310.

[10] Ipanaqué, R., Iglesias, A.: A Mathematica package for computing and visualizing the Gauss map of surfaces. *Lectures Notes in Computer Science,* **3482** (2005) 492-501.

[11] McCabe, W.L., Thiele, E.W.: Graphical design of distillation columns, *Ind. Eng. Chem.*, **17** (1925) 605.

[12] McCabe, W.L., Smith, J. C., Harriott, P.: *Unit Operations of Chemical Engineering,* Sixth Edition, McGraw-Hill, Boston (2001).

[13] McCallum, S.: Solving polynomial strict inequalities using cylindrical algebraic decomposition. *The Computer Journal,* **36**(5) (1993) 432-438.

[14] Strzebonski, A.: An algorithm for systems of strong polynomial inequalities. *The Mathematica Journal,* **4**(4) (1994) 74-77.

[15] Strzebonski, A.: Solving algebraic inequalities. *The Mathematica Journal,* **7** (2000) 525-541.