suma+ 60
Febrero 2009, pp. 7-20

Maxima, un sistema libre de cálculo simbólico y numérico

Dentro del contexto de las TIC aplicadas a la enseñanza de las matemáticas, se propone la introducción del sistema libre de cálculo simbólico Maxima, inicialmente desarrollado en el MIT. Maxima ofrece a estudiantes, profesores y profesionales un amplio conjunto de herramientas de cálculo, tanto simbólico como numérico, así como capacidades avanzadas de representación gráfica y un lenguaje de programación sencillo de aprender. También se incluyen ejemplos de actividades de aula reales.

Within the context of ICT applied to mathematics education, we propose the introduction of the Maxima computer algebra system, a free software project originally developed at MIT. Maxima is able to bring students, teachers and professionals a wide range of computing tools, both symbolic and numeric, together with strong plotting capabilities and an easy to learn programming language. Examples of real classroom activities are also included.

Introducción

Se presenta y describe el programa Maxima. Aunque habitualmente encuadrado en lo que se conoce como Sistemas de Cálculo Simbólico (o CAS, *Computational Algebra System*), Maxima es realmente un programa de matemáticas generales, pues junto a sus habilidades simbólicas hay que añadir las numéricas y gráficas, así como un lenguaje de programación propio que lo convierten en un entorno versátil y adaptable a todas las necesidades, tanto como calculadora personal, como herramienta pedagógica o instrumento de investigación.

Se configura este artículo en seis secciones: en la primera se aborda su historia, desde sus comienzos en el MIT hasta el momento actual como un proyecto libre alojado en los repositorios de Sourceforge; se pasa a continuación a mostrar algunos de los entornos de usuario más comunes, seguido de una sesión de trabajo en la que se hace un breve recorrido por las capacidades más destacables de este programa; continúa con una serie de ejemplos de aplicación del programa como herramienta pedagógica, para terminar con unas reflexiones finales y comentarios sobre recursos disponibles.

Historia de Maxima

A finales de los años sesenta, varias agencias gubernamentales norteamericanas, como el *U.S. Departament of Energy* (DOE), *National Aeronautics and Space Administration* (NASA), *Office of Naval Research* (ONR) y *U.S Air Force* (USAF), notaron la necesidad de disponer de un software matemático que les permitiera superar la dependencia que tenían respecto de programas exclusivamente numéricos, los cuales ya se venían utilizando desde la época de von Neumann y sus trabajos con el ENIAC, unos veinte años atrás, para la predicción meteorológica mediante la resolución de sistemas de ecuaciones diferenciales. La idea consistía en ver la manera de que las máquinas manipulasen las expresiones matemáticas de forma similar a como lo hacen los humanos.

Por esa época, en el MIT (*Massachussets Institute of Technology*) se presentó una tesis doctoral sobre integración simbólica cuyo autor era Joel Moses y que había sido supervisada por Marvin Minsky, uno de los gurús de lo que por aquella época ya se llamaba Inteligencia Artificial. Las agencias antes mencionadas aportaron los fondos necesarios y firmaron con el MIT un acuerdo para el desarrollo de este proyecto, al cual se le daría el nombre de Macsyma, (*MAC's SYmbolic MAnipulation system*), ya que MAC (*Machine Aided Cognition*) había sido el nombre original del equipo

Mario Rodríguez Riotorto
IES Punta Candieira de Cedeira. A Coruña

encargado de desarrollar el proyecto, más tarde conocido como Laboratory for Computer Science. En palabras del propio Marvin Minsky, Macsyma pretendía automatizar las transformaciones simbólicas que realizaban los matemáticos, a fin de entender la capacidad de los ordenadores para actuar de forma inteligente.

El proyecto arranca en 1969 y se desarrolla en lenguaje Lisp durante la década de los setenta. En 1982, ya finalizado el proyecto, una copia es transferida al DOE y otra a la empresa Symbolics Inc., que comercializaba las que se conocían como Lisp Machines. Symbolics se limitó a utilizar Macsyma como una demo para sus máquinas, pero nunca mostró interés en el desarrollo y mejora de este programa; tiempo después los derechos fueron vendidos a otra empresa, que aunque sí dedicó dinero y esfuerzos en la mejora del software, fue un fracaso comercial frente al auge que ya habían cobrado Mathematica y Maple, ambos inspirados en el propio Macsyma, y nacidos durante la década de los ochenta.

Pero hubo otra historia paralela. La copia que el MIT había enviado al Departamento de Energía, conocida como DOE-Macsyma, era utilizada por un reducido grupo de investigadores, que aunque no tenían permiso para distribuirla, sí les permitía trabajar sobre ella, depurar el código y actualizarlo al nuevo estándar Common Lisp. La figura clave de esta época fue William Schelter de la Universidad de Texas en Austin; fue él quien mantuvo viva la llama de DOE-Macsyma para la comunidad científica y fue también él quien en 1998 conseguiría el permiso del DOE para distribuirla bajo la *General Public License* de la *Free Software Foundation;* es decir, a partir de ahora Maxima, así renombrada para evitar problemas legales con la versión comercial, tendría su código abierto y pasaría a ser libre (para redistribuirlo, modificarlo, estudiarlo y mejorarlo). Poco tiempo después, en julio de 2001, Schelter fallece, pero su legado queda con nosotros. Sin él, dado el fracaso de la versión comercial, quizás la comunidad hubiese perdido irremisiblemente Maxima.

Hoy en día el proyecto Maxima se aloja en los repositorios de Sourceforge (Maxima Team, 2008a) y es mantenido por un grupo internacional de programadores cuyo interés no es otro que el de seguir mejorando este programa que por ser libre, paradójicamente, no es de nadie y es de todos. Desde su página web se puede acceder a toda la información relevante. El manual de referencia y diverso material introductorio también se encuentran en castellano (Maxima Team, 2008b, 2008c).

Entornos de ejecución

Maxima se puede ejecutar en Linux, Mac y Windows. Tanto el código fuente como algunos binarios precompilados se pueden descargar desde la página de proyecto[1].

El proyecto Maxima centra su atención en el desarrollo y mantenimiento de su motor matemático sin preocuparse en exceso sobre las interfaces de usuario, tarea que se deja para proyectos mantenidos por otros equipos de programadores. No obstante, los desarrolladores del programa mantienen dos interfaces de sofisticación mínima, uno de texto (Maxima) y otro semigráfica (Xmaxima); la Figura 1 es una captura del entorno de texto y la Figura 2 del entorno semigráfico. Como se ve, las expresiones matemáticas se construyen en ambos en base a caracteres ASCII; El entorno Xmaxima tiene la particularidad de disponer de un sencillo visor de páginas web en su ventana inferior que le permite acceder a la ayuda o a cualquier otro documento en formato HTML.

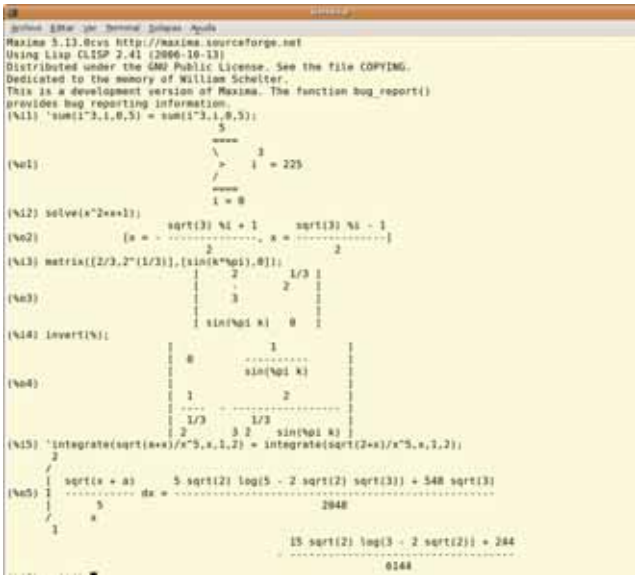

Figura 1

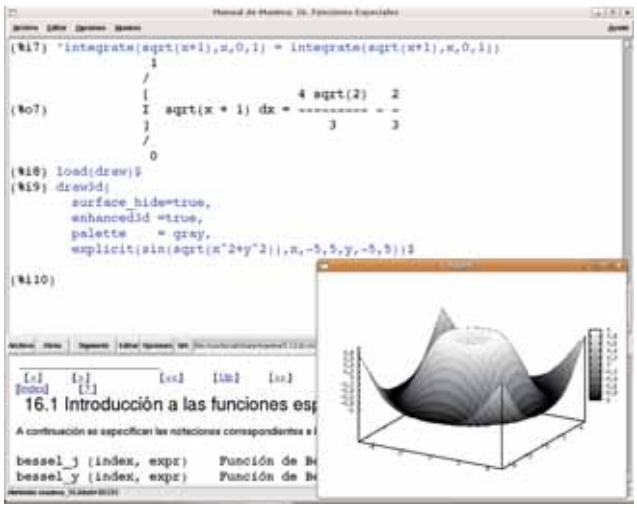

Figura 2

Probablemente el interfaz gráfico más utilizado sea Wxmaxima, que junto con los dos anteriores, son los que vienen incluidos en el ejecutable que se distribuye para usuarios de Windows. Es de más fácil uso que los dos anteriores, ya que junto a una mejor notación matemática, incorpora una serie de menús que facilitan el uso del programa; la Figura 3 es una captura de Wxmaxima en acción bajo Linux.

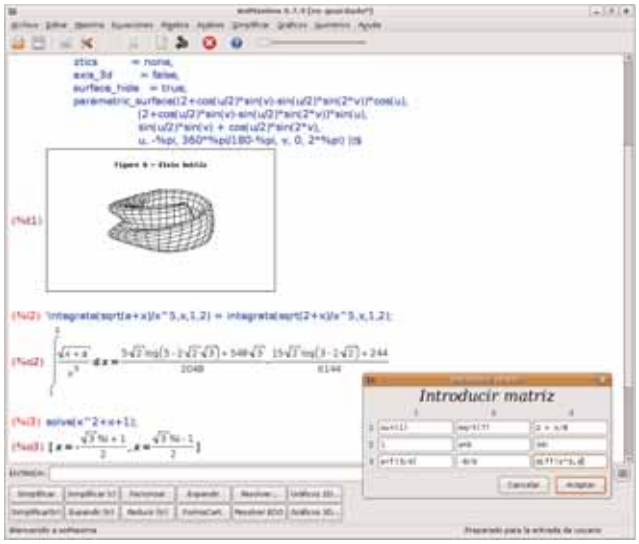

Figura 3

Un paso más en cuanto a nivel de sofisticación es el programa Texmacs, que se trata de un editor de textos WYSIWYG capaz de poder abrir sesiones de otros programas y formatear sus salidas a LaTex, con lo cual el nivel de presentación de expresiones matemáticas es óptimo, tal como se puede apreciar en la Figura 4.

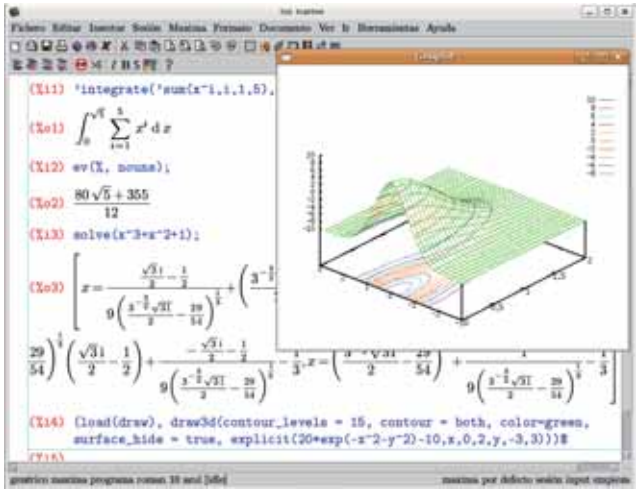

Figura 4

Existe también cierta actividad por parte de la comunidad de usuarios orientada al desarrollo de interfaces vía web, esto es, servidores de internet que tienen instalado Maxima y que permiten que un visitante que accede a la página pueda realizar cálculos, con cualquier navegador y desde cualquier equipo remoto. A éstos y a algunos otros entornos relacionados con Maxima se tiene acceso desde la página web del proyecto[2].

Una sesión con Maxima

Una vez se llama al programa veremos la cabecera, que incluye cierta información sobre la compilación y cómo ayudar al proyecto en caso de encontrar fallos:

```
Maxima 5.14.0cvs http://maxima.sourceforge.net
Using Lisp CLISP 2.41 (2006-10-13)
Distributed under the GNU Public License. See the file
COPYING.
Dedicated to the memory of William Schelter.
This is a development version of Maxima. The function
bug_report()
provides bug reporting information.
(%i1)
```

Tras la cabecera, el símbolo *%i1* nos indica que Maxima está esperando nuestra primera entrada (*i* de *input*). Pidámosle ahora la ejecución de un sencillo cálculo:

```
(%i1) (5/10 + 5!)^2;
(%o1)
```

$$\frac{58081}{4}$$

```
(%i2)
```

La instrucción debe terminarse con el punto y coma (*;*) antes de pulsar la tecla de retorno. La respuesta que se obtiene está etiquetada con *%o1* (*o* de *output*), indicando después que está a la espera de la segunda entrada por parte del usuario. Como se ve, Maxima evita dar resultados en formato decimal, pero se le pueden solicitar haciendo uso de la función *float*, tal como se indica a continuación, donde se calcula el valor de π en doble precisión:

```
(%i2) float(%pi);
(%o2)
                        3.141592653589793
```

Si se quiere mayor precisión, se asignará a la variable global *fpprec* el número de decimales deseado y se hará uso de la función *bfloat* (*b* de *big float*). En este ejemplo se ve que el operador a utilizar para asignar un valor a una variable son los dos puntos (*:*) y que cuando una instrucción termina con el

símbolo de dólar (*$*) Maxima omite el resultado calculado, de ahí que no aparezca *%o3*

```
(%i3) fpprec:200$    bfloat(%e);

(%o4)

2.71828182845904523536028747135266249775724709369\
99959574966967627724076630353547594571382178525l\
66427427466391932003059921817413596629043572900l\
34295260595630738132328627943490763233829880753l\
95251019b0
```

Junto con las constantes *%pi* y *%e* (la base de los logaritmos naturales) Maxima también conoce la razón áurea (*%phi*), la constante de Euler-Mascheroni (*%gamma*) y la unidad imaginaria (*%i*), la cual utilizaremos para construir números complejos. En los siguientes ejemplos autoexplicativos, las parejas de símbolos /* *y* */ encierran comentarios que el intérprete de instrucciones ignorará:

```
(%i5) z1: 3+4*%i; /* complejo en forma rectangular */
(%o5)
```

$$4i+3$$

```
(%i6) z2: exp(7*%pi/8*%i); /* forma polar */
(%o6)
```

$$e^{\frac{7i\pi}{8}}$$

```
(%i7) z1+z2;
(%o7)
```

$$e^{\frac{7i\pi}{8}}+4i+3$$

```
(%i8) rectform(%); /* pide forma rectangular */
(%o8)
```

$$i\left(\sin\left(\frac{7\pi}{8}\right)+4\right)+\cos\left(\frac{7\pi}{8}\right)+3$$

```
(%i9) polarform(%o7); /* pide forma polar */
(%o9)
```

$$\sqrt{\left(\sin\left(\frac{7\pi}{8}\right)+4\right)^2+\left(\cos\left(\frac{7\pi}{8}\right)+3\right)^2}\,e^{i\arctan\left(\frac{\sin\left(\frac{7\pi}{8}\right)+4}{\cos\left(\frac{7\pi}{8}\right)+3}\right)}$$

En los ejemplos anteriores, el símbolo *%* aislado hace referencia al último resultado devuelto por Maxima, mientras que *%o7* se refiere, obviamente, a la salida número siete.

En materia algebraica, Maxima puede expandir y factorizar expresiones, así como calcular valores numéricos de las misma,

```
(%i10) expand((a-b)^2*(a-1));
(%o10)
```

$$ab^2-b^2-2a^2b+2ab+a^3-a^2$$

```
(%i11) factor(%);
(%o11)
```

$$(a-1)(b-a)^2$$

```
(%i12) /* Valor numérico. ¡Ojo: se usa la igualdad! */
       subst([a=1/5, b=2], %);
(%o12)
```

$$-\frac{324}{125}$$

Existen varias funciones para la resolución de ecuaciones, siendo la más general *solve*. En el siguiente ejemplo se resuelve un sistema no lineal que incluye un coeficiente literal,

```
(%i13) solve([3*x^2-y=6,x^2=y+a],[x,y]);
(%o13)
```

$$\left[\left[x=\frac{-\sqrt{6-a}}{\sqrt{2}},y=-\frac{3a-6}{2}\right],\left[x=\frac{\sqrt{6-a}}{\sqrt{2}},y=-\frac{3a-6}{2}\right]\right]$$

Los corchetes delimitan listas cuyos elementos se separan por comas, por lo que los argumentos de *solve* son dos listas, la primera con las ecuaciones a resolver y la segunda con las incógnitas a aislar; el resultado que se obtiene es también una lista. Dado que Maxima está programado en Lisp (*List Processing*) no es de extrañar su habilidad para operar y manipular esta estructura de datos, con la cual, dicho sea de paso, se puede representar cualquier otro tipo de información estructurada. La siguiente secuencia muestra algunas de sus posibilidades,

```
(%i14) r: [1/2,[a,3],sqrt(3)/2,%pi];
(%o14)
```

$$\left[\frac{1}{2},[a,3],\frac{\sqrt{3}}{2},\pi\right]$$

```
(%i15) r[3]; /* devuelve el tercer elemento */
(%o15)
```

$$\frac{\sqrt{3}}{2}$$

```
(%i16) reverse(r); /* le da la vuelta a la lista */
(%o16)
```

$$\left[\pi,\frac{\sqrt{3}}{2},[a,3],\frac{1}{2}\right]$$

```
(%i17) r[2]: 0; /* cambia segundo elemento */
(%o17)
                              0
(%i18) r; /* estado actual de r */
(%o18)
```

$$\left[\frac{1}{2},0,\frac{\sqrt{3}}{2},\pi\right]$$

```
(%i19) map(sin, r); /* aplica seno a elementos */
(%o19)
```

$$\left[\sin\left(\frac{1}{2}\right),0,\sin\left(\frac{\sqrt{3}}{2}\right),0\right]$$

(%i20) apply("+", r); /* suma elementos */
(%o20)

$$\pi+\frac{\sqrt{3}}{2}+\frac{1}{2}$$

(%i21) 5*r; /* producto de número por vector */
(%o21)

$$\left[\frac{5}{2},0,\frac{5\sqrt{3}}{2},5\pi\right]$$

(%i22) r.[a,b,c,d]; /* producto escalar */
(%o22)

$$\pi d+\frac{\sqrt{3}c}{2}+\frac{a}{2}$$

(%i23) r+[a,b,c,d]; /* suma vectorial */
(%o23)

$$\left[a+\frac{1}{2},b,c+\frac{\sqrt{3}}{2},d+\pi\right]$$

Vemos que las propias listas se pueden interpretar como vectores y operar como tales.

En cuanto al cálculo infinitesimal, Maxima puede calcular límites, tal como muestran los siguientes ejemplos,

(%i24) limit(1/sqrt(x-1),x,inf); /* x tiende a infinito */
(%o24)

$$0$$

(%i25) limit(1/(x-1),x,1,plus); /* límite por la derecha */
(%o25)

$$\infty$$

El símbolo *inf* hace referencia al infinito y *minf* al menos infinito, de igual modo que plus se utiliza para calcular el límite lateral derecho y minus para el izquierdo. A continuación se aborda el cálculo de la derivada de una función:

(%i26) diff(x^log(a*x),x); /* primera derivada respecto de x */
(%o26)

$$x^{\log(ax)}\left(\frac{\log(ax)}{x}+\frac{\log(x)}{x}\right)$$

(%i27) diff(x^log(a*x),x,2); /* segunda derivada */
(%o27)

$$x^{\log(ax)}\left(\frac{\log(ax)}{x}+\frac{\log(x)}{x}\right)^2+x^{\log(ax)}\left(\frac{-\log(ax)}{x^2}-\frac{\log(x)}{x^2}+\frac{2}{x^2}\right)$$

y la integración,

(%i28) integrate(cos(x)^3/exp(x),x); /* integral indefinida */
(%o28)

$$\frac{e^{-x}\left(3\sin(3x)-\cos(3x)+15\sin(x)-15\cos(x)\right)}{40}$$

(%i29) integrate(cos(x)^3/exp(x),x,0,2); /* integral definida */
(%o29)

$$\frac{e^{-2}\left(3\sin(6)-\cos(6)+15\sin(2)-15\cos(2)\right)}{40}+\frac{2}{5}$$

La forma más directa de construir una matriz es mediante la función *matrix*, que admite como argumentos tantas listas como sea necesario, cada una de ellas representando una fila,

(%i30) A: matrix([2,-6,0],[4,6,-3],[0,4,0]);
(%o30)

$$\begin{pmatrix}2&-6&0\\4&-6&-3\\0&4&0\end{pmatrix}$$

(%i31) ident(3); /* matriz identidad de orden 3 */
(%o31)

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

(%i32) invert(A); /* inversión de A */
(%o32)

$$\begin{pmatrix}\frac{1}{2}&0&\frac{3}{4}\\0&0&\frac{1}{4}\\\frac{2}{3}&\frac{-1}{3}&\frac{3}{2}\end{pmatrix}$$

(%i33) A.%; /* producto matricial */
(%o33)

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

No nos extendemos más en mostrar las capacidades simbólicas y numéricas de Maxima, siendo imposible hacerle justicia a este programa en sólo unas pocas páginas. Sí se puede mencionar que aunque aquí se ha dado un breve esbozo en base a contenidos que resultan familiares a alumnos de nivel de secundaria, las habilidades de Maxima van mucho más allá de este nivel. No se ha hecho referencia alguna a los métodos numéricos de resolución de ecuaciones o de integración, ni a los procedimientos de cuadratura de ecuaciones diferenciales, a los módulos de probabilidad, simulación estocástica, estadística descriptiva e inferencial, álgebra lineal (vectores y valores propios, descomposiciones de Jordan), operaciones con conjuntos, combinatoria, teoría de grafos, análisis tensorial y un largo etcétera. El manual de referencia, traducido al español (Maxima Team, 2008b), es accesible desde el propio programa haciendo uso del operador *?*, tal como muestra el

siguiente ejemplo en el que se pide información sobre cierta función,

```
(%i34) ? sqrt

— Función: sqrt (<x>)
    Raíz cuadrada de <x>. Se representa internamente por
    `<x>^(1/2)'.
    Véase también `rootscontract'.

    Si la variable `radexpand' vale `true' hará que las raíces
    `n'-ésimas de los factores de un producto que sean
    potencias de `n' sean extraídas del radical; por ejemplo,
    `sqrt(16*x^2)' se convertirá en `4*x' sólo si `radexpand'
    vale `true'.
```

Cuando Maxima se carga en la memoria, no todas sus funciones son accesibles, ya que muchas se encuentran en módulos o paquetes adicionales que se distribuyen junto con el núcleo del programa. Un ejemplo de esto es el paquete *draw*, un interfaz para la generación de gráficos en dos y tres dimensiones3. La siguiente secuencia carga el paquete con la llamada a la función *load* y luego genera una escena en dos dimensiones, que incluye una función explícita, otra paramétrica y un conjunto de puntos aislados generados aleatoriamente; el resultado en la Figura 5.

```
(%i35) load(draw)$
(%i36) draw2d(
        key          = "Polinomio cubico",
        explicit(%pi*x^3+sqrt(2)*x^2+10,x,0,2),
        color        = blue,
        key          = "Curva parametrica",
        line_width   = 3,
        nticks       = 50,
        parametric(2*cos(rrr)+3, rrr, rrr, 0, 6*%pi),
        line_type    = dots,
        points_joined = true,
        point_type   = diamant,
        point_size   = 3,
        color        = red,
        line_width   = 1,
        key          = "Datos empiricos",
        points(makelist(random(40.0),k,1,5)),
        title        = "REPRESENTACION DE CURVAS",
        terminal     = eps )$
```

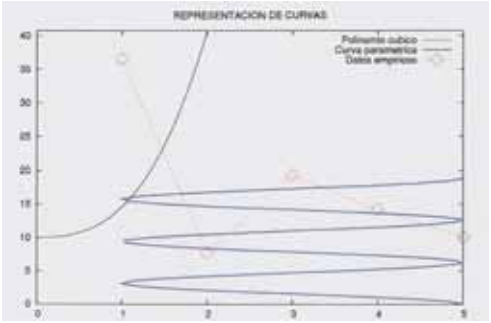

Figura 5

Los argumentos de *draw2d* son de dos tipos: *opciones*, los que tienen igualdades, y *objetos gráficos*, cuyas sintaxis son similares a las funciones. Las opciones se leen secuencialmente y una vez se les asignan valores éstos se mantienen activos hasta que se vuelven a cambiar o hasta el final de la descripción de la escena. Algunas opciones afectan a cómo se dibujarán los objetos y otras hacen referencia al gráfico en su conjunto.

La siguiente instrucción genera un gráfico tridimensional consistente en una superficie explícita y dos curvas paramétricas. El resultado en la Figura 6.

```
(%i37) draw3d(
        color        = green,
        explicit(exp(sin(x)+cos(x^2)),x,-3,3,y,-3,3),
        color        = blue,
        parametric(cos(5*u)^2,sin(7*u),u-2,u,0,2),
        color        = brown,
        line_width   = 2,
        parametric(t^2,sin(t),2+t,t,0,2),
        surface_hide = true,
        title        = "Superficies y curvas",
        color        = red,
        label(["ARRIBA",-2,0,3]),
        label(["ABAJO",2,0,-3]),
        rot_horizontal = 10,
        rot_vertical = 84,
        terminal     = eps )$
```

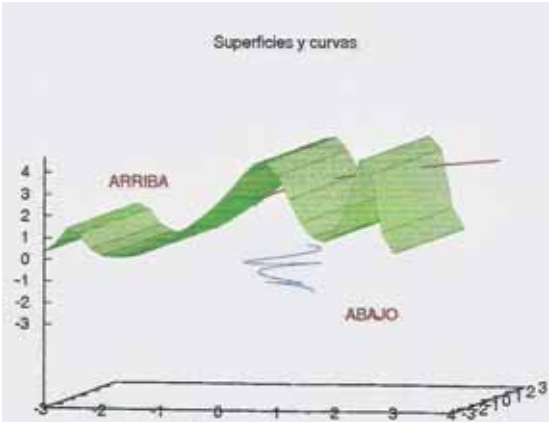

Figura 6

La opción gráfica *terminal* es la que se encarga de seleccionar el formato del gráfico; se admiten eps, png, jpg, gif y gif animado, siendo su valor por defecto la ventana gráfica de la pantalla[4].

Otra de las características de Maxima es un lenguaje de programación sencillo que incluye los elementos básicos de declaración de variables locales, diferentes tipos de bucles y condicionales. A modo de ejemplo, el siguiente código define una función de nombre *fact* que admite un único argumento, que debe ser entero no negativo, del que devuelve su factorial. El operador que permite definir la función es := , al que le

sigue un condicional que controla si el argumento es entero no negativo; si la condición se verifica, entonces se crea un bloque con variable local *prod* inicializada a 1 y que irá recogiendo los sucesivos productos; terminado el bucle, el valor *prod* será el mostrado al usuario. De evaluarse el condicional como falso, entonces se emitirá un mensaje de error.

```
(%i38) fact(n):=
       if integerp(n) and n >= 0
         then block([prod:1],
              for k:1 thru n do prod:prod*k,
              prod)
         else error("Ojo: ", n, "no es entero no-negativo") $
```

Una vez definida la función, podemos hacer algunos cálculos:

```
(%i39) fact(-6);
Ojo:  - 6 no es entero no-negativo
#0: fact(n=-6)
 — an error.  To debug this try debugmode(true);

(%i40) fact(0);
                                1
(%i41) fact(200);

788657867364790503552363213932185062295135977687\
173263294742533244359449963403342920304284011984\
623904177212138919638830257642790242637105061926\
624952829931113462857270763317237396988943922445\
621451664240254033291864131227428294853277524242\
407573903240321257405579568660226031904170324062\
351700858796178922227896237038973747200000000000\
000000000000000000000000000000000000000000
```

Este ejemplo sirve también para mostrar que Maxima no pone límites al número de dígitos de los enteros. Valga también decir que esta función no era necesaria, pues ya se dispone de la función factorial nativa compilada (*200!;*). En la definición de funciones se admite la recursividad y la admisión de un número arbitrario de argumentos, así como que éstos puedan ser funciones, además de expresiones o nombres de variables. La práctica habitual es que el usuario escriba su propio código en un fichero de texto de extensión *.mac* y que una vez dentro de Maxima lo cargue ejecutando la instrucción *load("ruta/fichero.mac")*.

Por último, para los más intrépidos, téngase en cuenta que el hecho de que Maxima sea libre implica que su código fuente en Common Lisp es accesible para el usuario, lo que a su vez conlleva la posibilidad de escribir directamente los propios paquetes en este lenguaje, lo que en determinados contextos permitirá generar rutinas más rápidas. En todo caso, no es necesario conocer Lisp a fin de sacarle partido al programa, pudiendo implementar cualquier procedimiento en lenguaje Maxima.

Maxima como recurso pedagógico

En general, se pueden considerar tres contextos diferentes en los que un programa de estas características puede resultar útil: investigación, matemática aplicada y docencia.

Sin duda es en el campo de la investigación donde un programa de matemáticas generales tenga menos peso, ya que campos especializados requieren de software específico; excepción evidente es el campo del álgebra computacional, donde Maxima, al ser de código abierto y estar programado en un lenguaje especialmente adaptado al procesamiento simbólico, aventaja a otros programas similares de carácter comercial, en los que el código fuente queda oculto al usuario o, en este caso, investigador.

En un contexto más pragmático, que es el de la matemática aplicada, Maxima gana en importancia; no en vano algunos de sus programadores actuales son ingenieros que desarrollan su actividad profesional en el ámbito industrial. Maxima puede superar con ventaja otras herramientas matemáticas a las que los ingenieros están habituados, como las hojas de cálculo o programas de cálculo numérico específico con ninguna o escasa capacidad de procesamiento simbólico. Maxima ha importado potentes rutinas numéricas mediante la traducción a Lisp de programas libres escritos originariamente en Fortran y bien conocidos y puestos a prueba durante muchos años (*quadpack* para integración numérica o *lapack* para álgebra lineal son un par de ejemplos).

Pero sin duda el campo en el que un programa libre como Maxima tiene pocos rivales es el de la formación y docencia. Ningún docente puede sugerir a sus alumnos que intenten disponer de programas como Mathematica o Maple en sus equipos (¿no sería ésta una invitación encubierta a la piratería?), y pocos afortunados podrán sugerir a sus administradores la compra de múltiples licencias para dotar a sus laboratorios y aulas de informática con estos u otros programas de pago. El libre acceso a un programa de cálculo simbólico, numérico y gráfico puede romper esquemas en lo que al uso de las TIC en la enseñanza matemática se refiere. Junto con Maxima, otros programas de cálculo simbólico y código abierto son Axiom o SymPy.

Se presentan a continuación algunas ideas sobre el uso de Maxima como herramienta pedagógica. La mayor parte de ellas han formado parte de nuestra actividad docente en secundaria, bien por parte del autor, bien por parte de otros colegas.

Ejemplo 1

Esta actividad está orientada a alumnos de tercero de ESO. Se trata de ir resolviendo ecuaciones paso a paso, haciendo énfa-

sis en las operaciones que hay que hacer cada vez; la idea es romper con la clásica dinámica de "lo que está sumando pasa restando", etc. El diálogo se efectúa en inglés porque el grupo para el que se escribió era una sección bilingüe (enseñanza CLIL); el cambio de idioma implica únicamente abrir el fichero texto que guarda el programa[5] e ir traduciendo las cadenas alfanuméricas correspondientes. Las entradas que realiza el alumno se han marcado con -> para facilitar la interpretación de la sesión; tales entradas deben ir seguidas de una pulsación de la tecla de retorno:

```
(%i1) load(ec1)$ /* carga fichero con programa */

=== WELCOME TO FIRST DEGREE EQUATIONS ===

Exercise number ... (max. 4)
->  1 ;
The unknown of this equation is:  x

                        3x+1=7x

 +: Add an expression
 -: Subtract an expression
 *: Multiply by an expression
 /: Divide by an expression
 p: Remove parentheses
 f: Factorize
Your choice is ...
->  +;
Write number or algebraic expression:
->  -1;
                        3x=7x-1

 +: Add an expression
 -: Subtract an expression
 *: Multiply by an expression
 /: Divide by an expression
 p: Remove parentheses
 f: Factorize
Your choice is ...
->  -;
Write number or algebraic expression:
->  7*x;

                        -4x=-1

 +: Add an expression
 -: Subtract an expression
 *: Multiply by an expression
 /: Divide by an expression
 p: Remove parentheses
 f: Factorize
Your choice is ...
->  /;
Write number or algebraic expression:
->  -4;

Congratulations! You did it:

                        x = 1/4
————————————————————
Exercise number ... (max. 4); 0 => Exit
->  0;
```

En esencia, se trata de un bucle que presenta de cada vez un menú con las acciones que el alumno puede realizar; si éste realiza una acción que no sea la óptima, como por ejemplo dividir por -5 en lugar de -4 en el último paso, Maxima se lo admitirá, pero tendrá que arreglar la situación haciendo únicamente uso de las opciones del menú. Para clarificar, veamos cómo hubiese sido el diálogo en este caso:

```
 +: Add an expression
 -: Subtract an expression
 *: Multiply by an expression
 /: Divide by an expression
 p: Remove parentheses
 f: Factorize
Your choice is ...
->  /;
Write number or algebraic expression:
->  5;
```

$$-\frac{4x}{5}=-\frac{1}{5}$$

```
 +: Add an expression
 -: Subtract an expression
 *: Multiply by an expression
 /: Divide by an expression
 p: Remove parentheses
 f: Factorize
Your choice is ...
->  *;
Write number or algebraic expression:
->  -5/4;

Congratulations! You did it:
```

$$x=\frac{1}{4}$$

Aquí se ha mostrado una sencilla ecuación de primer grado, pero la idea se puede extender a segundo grado (el alumno tratará de formar cuadrados de binomios y extraer la raíz cuadrada), sistemas de ecuaciones e incluso ecuaciones con coeficientes literales, no numéricos. Nótese que en ningún momento se ha hecho uso de las rutinas de resolución de ecuaciones de Maxima.

Ejemplo 2

Cuando en una sesión de álgebra en secundaria se hace necesario trabajar con problemas contextualizados, se podrán ensayar más problemas si la parte mecánica de resolución de la ecuación resultante se deja a la máquina; así, para cada problema, la secuencia de la tarea a realizar por parte del alumno sería:

1. Lectura, interpretación y extracción de los datos necesarios para la resolución del problema.
2. Diseño del modelo matemático y planteamiento de la ecuación o sistema.
3. Resolución con Maxima de la ecuación o sistema.
4. Interpretación y validez de las soluciones y eliminación de las no válidas dentro del contexto dado.

El hecho de que el paso número 3 se deje a la máquina permitirá un ahorro de tiempo que se podrá dedicar a realizar más problemas y trabajar con más intensidad el resto de los apartados. Huelga decir que esto no exime al alumno de saber resolver ecuaciones manualmente, pero en esta sesión se trabaja más la matematización de situaciones contextualizadas. Al contrario que en el ejemplo anterior, aquí sí que será necesario hacer uso de las rutinas de Maxima para resolver ecuaciones, principalmente *solve*. Sigue un ejemplo de enunciado y su resolución.

> El tratado matemático chino más antiguo del que se tiene noticia es el Suan shu shu, encontrado en unas excavaciones realizadas durante el año 1983 en la tumba de un oficial administrativo enterrado en el año 186 AC (Dinastía Han Occidental). El documento está escrito en 190 tiras de bambú y consta de una amplia colección de problemas aritméticos. Este problema es el número 51 de la serie: "En un reparto, si a cada persona se le dan dos monedas, entonces sobran tres monedas; si a cada persona se le dan tres monedas, entonces faltan dos monedas. ¿Cuántas personas y monedas hay?"

Si llamamos p al número de personas y m al de monedas, podemos plantear el modelo como:

$$2p = c - 3$$
$$3p = c + 2$$

o como:

$$2p + 3 = c$$
$$3p - 2 = c$$

En cualquier caso, Maxima resuelve. Los alumnos podrán comprobar rápidamente qué modelos de los planteados por ellos son equivalentes.

```
(%i1) solve([2*p=c-3, 3*p=c+2], [p,c]);
(%o1)
```

$$[[p = 5, c = 13]]$$

```
(%i2) solve([2*p+3=c, 3*p-2=c], [p,c]);
(%o2)
```

$$[[p = 5, c = 13]]$$

Ejemplo 3

Cuando un Departamento Didáctico decide incluir en su programación que sus alumnos adquieran a lo largo de su formación secundaria unas mínimas habilidades en técnicas de programación, no es difícil que puedan llegar al último curso de Bachillerato con el bagaje necesario para afrontar determinadas tareas de desarrollo de algoritmos numéricos con el lenguaje de Maxima, siempre que haya un planteamiento conjunto entre los miembros del Departamento. Tal objetivo se puede conseguir si ya en la ESO utilizan Maxima a modo de calculadora, para luego trabajar a un nivel elemental condicionales y bucles, al tiempo que escriben sus primeras funciones. Algunos bloques del currículo del Bachillerato se prestan muy bien a que luego se ejerciten estas habilidades. De una unidad didáctica de la asignatura optativa de segundo de Bachillerato Métodos Estadísticos y Numéricos se han extraído los siguientes problemas (la Figura 7 que acompaña al enunciado del problema también se ha generado con las rutinas gráficas de Maxima[6]):

Problema 1. Sea una función *f(x)* a integrar numéricamente en el intervalo [*a*, *b*] haciendo uso de n rectángulos de igual base,

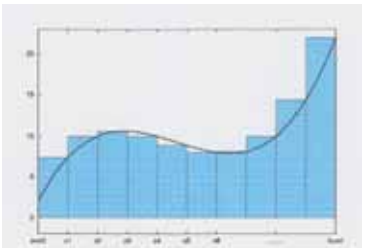

Figura 7

a) ¿Cuánto mide la base d de los rectángulos?
b) Supón que llamamos x_i a los sucesivos puntos de la partición, tal como muestra el gráfico adjunto, ¿cómo expresarías el área del primer rectángulo? ¿del segundo? ¿del último? ¿del i-ésimo rectángulo?
c) ¿Cómo expresarías x_i en función de a y de d?
d) Haciendo uso de los resultados de los apartados anteriores, completa la siguiente expresión:

$$\int_a^b f(x)\,dx \approx \sum_{\cdots\cdots}^{\cdots\cdots} \cdots\cdots\cdots\cdots$$

Problema 2. La programación de un integrador numérico pasa por escribir un código que realice las operaciones deducidas en el problema anterior. Trata de escribir tú mismo este código atendiendo a las siguientes especificaciones:

- El nombre de la función a programar y sus argumentos deben ser *intnum(fun,var,a,b,n)*, donde *fun* es el integrando, *var* es el nombre de la variable independiente,

a y *b* los límites del intervalo de integración y n el número de rectángulos.
- Utiliza un *block* que te permita especificar las variables locales que vayas a necesitar.
- Una de las variables, de valor inicial nulo, deberá ir guardando las sumas de las áreas.

Siguen en esta unidad algunos otros problemas en los que se invita al alumno a poner a prueba su algoritmo comparándolo con integrales definidas de fácil resolución manual y comprobar cómo varía la precisión alterando el parámetro *n*. También se le pedirá la integración de funciones que ni él ni Maxima podrán resolver de forma exacta, así como diseñar variantes como que la altura de los rectángulos se evalúe en los puntos medios de los intervalos de la partición o la sustitución de rectángulos por trapecios.

En definitiva, de lo que se trata es de llegar a una expresión como para a partir de ella conseguir un programa similar a éste:

```
intnum(fun,var,a,b,n):=
  block([suma: 0, d: (b-a) / n],
    for i:1 thru n do
      suma: suma + subst(var = a+i*d, fun),
    float(d * suma) )$
```

y realizar cálculos tales como

```
(%i1) intnum(u^2,u,0,1,100);
(%o1)                          0.33835
```

Ejemplo 4

En el sistema educativo alemán existe una prueba al final de la formación secundaria similar a nuestra Selectividad. A diferencia de la nuestra, tal prueba, allí llamada *Abitur*, es propuesta por los propios docentes de secundaria a sus alumnos. El siguiente problema ha formado parte de la *Abitur* presentada a los alumnos de un Instituto (*Gymnasium*) de la ciudad de Aquisgrán (*Aachen*) en el año 2006, la cual debían realizar frente a un ordenador con Maxima instalado y un procesador de textos para escribir y presentar sus resultados[7]:

El gráfico de la función coseno hiperbólico recibe el nombre de catenaria, pues su forma se ajusta a la de una cadena que cuelga sujeta por sus extremos y expuesta a la atracción gravitatoria. Haciendo uso de cierto parámetro positivo del cual dependerá la forma de la curva, tenemos :

$$f(x)=\cosh(x/a)$$

a) Obtén las representaciones gráficas de la catenaria para a = 1, 2, 3 y 4 en el dominio [-8, 8] y rango [0, 12] para las ordenadas. Añade para cada función una leyenda que haga referencia al valor paramétrico utilizado.

b) Demuestra que para cualquier a positivo, el punto más bajo de la catenaria se encuentra en el punto T(0, a).

En primer lugar se espera que el alumno defina en Maxima la función correspondiente y haga una representación gráfica como la siguiente, cuyo aspecto se muestra en la Figura 8.

```
(%i1) f(a,x):= a*cosh(x/a)$
(%i2) load(draw)$
(%i3) draw2d(
        yrange  = [0, 12],
        grid    = true,
        terminal = eps,
        key = "a = 1",  color = black, explicit(f(1,x), x, -8, 8),
        key = "a = 2",  color = red  , explicit(f(2,x), x, -8, 8),
        key = "a = 3",  color = blue , explicit(f(3,x), x, -8, 8),
        key = "a = 4",  color = green, explicit(f(4,x), x, -8, 8) )$
```

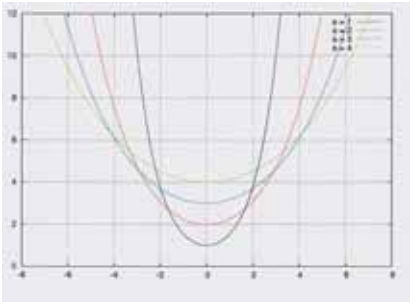

Figura 8

La respuesta a la segunda pregunta requiere echar mano de las capacidades simbólicas de Maxima, calculando la derivada de la función, igualándola a cero y resolviendo la ecuación resultante:

```
(%i4) diff(f(a,x), x)=0;
(%o4)
```

$$\sinh\left(\frac{x}{a}\right)=0$$

```
(%i5) solve(%,x);
(%o5)
                              [x=0]
```

Luego las coordenadas del punto crítico son, para un a>0 arbitrario,

```
(%i6) [0,f(a,0)];
(%o6)
                              [0, a]
```

Sólo falta comprobar ahora que la segunda derivada de la catenaria es positiva en este punto, para asegurar que el punto crítico corresponde a un mínimo local:

```
(%i7) diff(f(a,x),x,2);
(%o7)
```

$$\frac{\cosh\left(\frac{x}{a}\right)}{a}$$

```
(%i8) subst(x=0, %); /* evaluando en x=0 */
(%o8)
```

$$\frac{1}{a}$$

que obviamente es positivo al serlo el parámetro *a*.

El problema propuesto tiene más apartados, los cuales transcribimos a continuación:

c) Describe brevemente una interpretación del parámetro a de la función de la catenaria.
d) Un arco en forma de catenaria invertida es el Gateway Arch en St. Louis, Missouri, EEUU. Su perfil externo tiene una altura de 192 m y en el suelo una anchura de 192 m. El perfil interno tiene una altura de 187.50 m y un ancho a nivel del suelo de 161.50 m. La función de una catenaria invertida es $arch(x) = -acosh(x/a) + a + h$. Demuestra que su punto más alto se encuentra en HP(0, h).
e) Calcula los parámetros *a* y *h* de los perfiles externo e interno del Gateway Arch. El parámetro a debe calcularse numéricamente; utiliza para ello una de las funciones de Maxima para el efecto, tomando como solución inicial 1.0.
f) Dibuja los perfiles del Gateway Arch. Utiliza el dominio [-100, 100] y el rango [0, 200] para las ordenadas.
g) Calcula la superficie limitada por el perfil interior y el suelo.
h) Calcula nuevamente el parámetro a del perfil externo, pero ahora realiza tú mismo cada iteración del método de Newton. Resuelve la ecuación:

$$-acosh(96/a) + a + 192 = 0$$

comenzando con la solución inicial a = 30,0. Comprueba tu resultado con el del apartado e).

En el apartado e) el alumno debe hacer uso de la función *newton* que ejecuta el algoritmo de Newton-Raphson y que ya viene incluida en Maxima. En el apartado g) deberá utilizar la función *integrate* que ya conocemos.

Ejemplo 5

En este ejercicio se plantea el uso de Maxima para demostrar ciertos resultados de carácter teórico.

Sean $X_1 \sim N(m_1, s_1)$ y $X_2 \sim N(m_2, s_2)$ dos variables aleatorias normales independientes; sea también $Z = X_1 + X_2$. Demuestra que Z es también una v.a. normal y calcula sus parámetros.

Para resolver el problema tenemos en cuenta que la función característica de *Z* (la transformada de Fourier de su función de densidad) es igual al producto de las funciones características de X_1 y X_2.

```
(%i1) /* Función densidad de una normal N(m,s) */
      f(x,m,s):= 1 / (s*sqrt(2*%pi)) * exp(- (x-m)^2/(2*s^2))$
(%i2) /* Función característica de una normal N(m,s) */
      phi(t,m,s):= integrate(f(x,m,s)*exp(%i*t*x),x,minf,inf)$
(%i3) /* En general, para una N(m,s)... */
      (assume(s>0), phi(t,m,s));
(%o3)
```

$$e^{\frac{2imt - s^2t^2}{2}}$$

Se calcula el producto de las funciones características de X_1 y X_2 y se reduce el resultado; comparando éste con la salida *%o3* parece claro que estamos ante la función característica de una normal:

```
(%i4) (assume(s1>0, s2>0), phi(t,m1,s1) * phi(t,m2,s2));
(%o4)
```

$$e^{\frac{2im_2t - s_2^2t^2}{2} + \frac{2im_1t - s_1^2t^2}{2}}$$

```
(%i5) radcan(%);
(%o5)
```

$$e^{\frac{\left(s_2^2 + s_1^2\right)t^2 + \left(-2im_2 - 2im_1\right)t}{2}}$$

Una sencilla llamada a la función solve nos dirá qué valores corresponden a la esperanza y desviación típica de *Z*:

```
(%i6) solve([realpart(log(%o3))=realpart(log(%o5)),
            imagpart(log(%o3))=imagpart(log(%o5))],
           [m,s]);
(%o6)
```

$$\left[\left[m = m_2 + m_1, s = -\sqrt{s_2^2 + s_1^2}\right], \left[m = m_2 + m_1, s = \sqrt{s_2^2 + s_1^2}\right]\right]$$

Evidentemente, desechamos la primera solución, en la que $s<0$.

Reflexiones finales

Como se ha visto, los sistemas de cálculo simbólico pueden utilizarse como herramienta pedagógica en todos los niveles de la enseñanza. La propia experiencia, así como el intercambio de impresiones con otros colegas que también hacen uso de este y otros programas similares, permiten extraer algunas conclusiones, por supuesto opinables y posiblemente sujetas a no pocas matizaciones (García et al., 1995).

Como ventajas, destaca la flexibilidad que un sistema programable tiene a la hora de amoldarlo a multitud de situaciones, niveles y necesidades. Sistemas de cálculo como Maxima o

similares permiten experimentar directamente sobre modelos reales, de mayor dificultad matemática, evitando la dispersión de esfuerzos en cálculos tediosos y centrando la atención más en los conceptos que en las habilidades manipulativas, reforzando lo creativo frente a lo rutinario, permitiendo dedicar más tiempo a reflexionar sobre los modelos y los resultados. Las capacidades gráficas ayudan a interpretar los aspectos geométricos de los fenómenos bajo estudio; los gráficos son inmediatos, al contrario que cuando se hacen a mano en la pizarra, permitiendo la interactividad cambiando parámetros, tranformando las funciones, etc. Muy importante, la licencia libre de Maxima permite que los alumnos puedan disponer de él en casa, pudiendo diseñarse nuevas formas de trabajo personal y colaborativo. Otras materias que hacen uso de la matemática pueden también beneficiarse de él, incluso permitiendo el desarrollo de proyectos interdisciplinares; en particular, el tema de programación en las tecnologías de la información y la comunicación (TIC) es otro excelente ámbito de aplicación de Maxima, ya que al disponer de un lenguaje muy sencillo e intuitivo, el alumno que tenga que programar algoritmos lo puede hacer más fácil, ir al grano, sin preocuparse de declaraciones de variables, herencias de clases, crear otras estructuras de datos previas, etc., con el valor añadido de que el alumno aprende a controlar un programa que le será útil en su formación superior y durante su futuro profesional, no siendo pocos los departamentos universitarios y escuelas técnicas que están migrando a ésta y otras herramientas de código abierto, como Octave o R. Finalmente, el uso de software libre no es ajeno a la educación en valores. El alumno debe ser informado de que esto va más allá de la simple gratuidad del programa; lo que realmente hay detrás es la idea de que el conocimiento científico es un legado al que todo el mundo tiene derecho: a usarlo, a estudiarlo, a mejorarlo y a participar en su desarrollo.

Los sistemas de cálculo simbólico pueden utilizarse como herramienta pedagógica en todos los niveles de la enseñanza

Como en todo, también aquí hay inconvenientes. Aunque parece que cada vez menos, no siempre existe una disponibilidad razonable de los recursos informáticos y de hardware. Por otro lado, el docente que quiera hacer uso de estas herramientas, tendrá que reflexionar sobre la confección de nuevas formas de elaboración de contenidos y de evaluación de alumnos, tendrá que aprender el uso de una nueva herramienta, con su lenguaje de programación, que siendo sencillo y fácil de aprender sin mucho esfuerzo, le permitirá adaptar la herramienta a su forma de trabajo. Sin duda hay un trabajo extra por parte del profesor que quizás en algunos casos no se valore suficientemente a nivel administrativo.

Sistemas de cálculo como Maxima o similares permiten experimentar directamente sobre modelos reales, de mayor dificultad matemática

En cuanto al alumnado, existen ciertos riesgos, como que éste abandone sus esfuerzos por adquirir habilidades y automatismos manuales, o una pérdida del sentido crítico, materializándose en una fe ciega en los resultados del ordenador. Todo programa de ordenador tiene sus límites; de vez en cuando es bueno llevar al alumno hasta estos límites a fin de que experimente por qué no debe uno fiarse en exceso de una máquina. Una regla de oro: evitar convertir una clase de Matemáticas en una clase de Maxima; los errores sintácticos frenan la marcha de la clase; es imprescindible que los alumnos tengan que escribir al teclado lo menos posible durante la clase, a lo que ayudan en este aspecto los interfaces gráficos como Wxmaxima, cuyos menús permiten realizar un gran número de cálculos..., pero con cuidado, ya que hemos observado que los entornos gráficos pueden dispersar la atención del alumno (menús, opciones, botones, tipos de fuentes, colores, ventanas, más ventanas). Pero en fin, nada que una buena planificación de la clase no pueda solventar; en todo caso, la pauta a seguir será, como en todo lo que se empieza, la sabia estrategia de la prueba y el error, evitando tropezar dos veces en la misma piedra.

Comentarios a la bibliografía y enlaces

Desde que nace Macsyma hasta el momento actual, el programa ha pasado por varias manos y han existido, como ya queda comentado más arriba, dos historias paralelas. De cada una de estas etapas, han quedado registros escritos en forma de manuales, artículos científicos, tesis doctorales y *Technical Reports*. Algunas de las referencias que se comentan, y otras a las que aquí no se hace referencia, no son fáciles de conseguir y otras van pasando de mano en mano por la vía del escaneado y el correo electrónico. Aún así he decidido incluirlas aquí por su interés histórico y porque sus títulos sugieren el alcance y difusión que este software de cálculo simbólico tuvo durante las décadas de los ochenta y noventa, antes de la caída de la versión comercial y su renacimiento en la versión libre de Maxima.

Durante la etapa de su gestación en el MIT se han producido multitud de documentos alrededor de Macsyma, no sólo *Technical Reports* para consumo interno, sino también para el

público en general (MathLab Group, 1983; Fateman, 1977). Cuando Macsyma cambia de manos, la empresa que lo adquiere también generará documentación (Bogen, 1986; Symbolics, 1988) y lo mismo hará la siguiente, y actual propietaria, cuyo nombre coincide con el del programa (Macsyma, 1995; Jones et al., 1997).

La licencia libre de Maxima permite que los alumnos puedan disponer de él en casa, pudiendo diseñarse nuevas formas de trabajo personal y colaborativo

A lo largo de dos décadas, el uso de Macsyma en diferentes ámbitos ha sido constante, hasta que ha ido decayendo debido a una dudosa gestión empresarial y a la presión ejercida por Mathematica y Maple. Se citan algunas de las numerosas referencias existentes en la literatura, como su uso en el análisis estadístico (Heller, 1991), en Física (Ari, 1989), en la educación (Bratcher, 1989) y otros contextos (Delest, 1991; Clarkson, 1989; Ivie, 1978; Kwok, 1991; DeLoatch, 1987). También por su interés histórico, merece ser destacado el uso que Stephen Wolfram (1979), *alma mater* del Mathematica, hizo de Macsyma en relación a los diagramas de Feynman; además, entre Richard Feynman y él hubo un intercambio de cartas en las que ambos discutían sobre las posibilidades de Macsyma como herramienta de cálculo en física de partículas (Feynman, 2006).

Con el nacimiento del nuevo milenio, Macsyma languidece y Maxima toma la alternativa. Buena parte de las referencias citadas contienen material de indudable interés para la versión libre, aunque los dos programas no son completamente compatibles. Aunque los dos mantienen en común buena parte del núcleo simbólico desarrollado en el MIT, a partir del momento en el que se produce la bifurcación, las filosofías de desarrollo cambian y a iguales problemas se les dan soluciones diferentes; quizás esta diferencia se acentúa cada vez más en el momento presente, dado el dinamismo que está adquiriendo la versión libre.

Desgraciadamente, las referencias bibliográficas impresas relativas a Maxima aún están pendientes de publicación. El material existente circula en formatos electrónicos (esto es, html y pdf) a través de la red, probablemente a tenor de los tiempos que vivimos. La fuente fundamental de información es la página del proyecto (Maxima Team, 2008a), desde donde se puede acceder a la documentación más relevante, como el Manual de Referencia (Maxima Team, 2008b) y diversos tutoriales (Maxima Team, 2008c), algunos de ellos también en español. En estas páginas se encuentran enlaces a otros sitios en los que se hace uso de Maxima, especialmente como herramienta educativa, y en las que el lector encontrará abundante información adicional.

Abundando en el aspecto formativo, cualquier otro documento (libro, unidad didáctica, etc.) dedicado al uso de los sistemas de cálculo simbólico, no necesariamente Maxima, dará ideas y sugerencias sobre su uso en el aula (García et al., 1995).

NOTAS

1 http://sourceforge.net/project/platformdownload.php?group_id=4933

2 Un entorno muy versátil y que también utiliza salidas en formato LaTex es el editor Emacs, de amplio uso en sistemas Unix y también muy utilizado por usuarios de Maxima en Mac. Todo lo relativo a su instalación y configuración se puede consultar en la página:
http://members3.jcom.home.ne.jp/imaxima

3 Common Lisp no incluye rutinas gráficas, por lo que Maxima necesita un programa externo que realice los gráficos, que en este caso es Gnuplot. El ejecutable de instalación de Windows ya lo incluye, pero los usuarios de Linux y Mac deberán comprobar que tienen instalada la versión 4.2 de Gnuplot como mínimo.

4 Una amplia galería de las posibilidades gráficas del paquete draw, junto con los códigos correspondientes, se pueden ver en:
http://www.telefonica.net/web2/biomates/maxima/gpdraw

5 http://www.telefonica.net/web2/biomates/maxima/ec1.mac

6 El código que ha generado este gráfico es:

```
load(draw)$

f(x):=x^3-x^2-2*x+10$

draw2d(
  yrange=[-2,23],
  fill_color=cyan,
  rectangle([-2,f(-1.5)],[-1.5,0]),
  rectangle([-1.5,f(-1)],[-1,0]),
  rectangle([-1,f(-0.5)],[-0.5,0]),
  rectangle([-0.5,f(0)],[0,0]),
  rectangle([0,f(0.5)],[0.5,0]),
  rectangle([0.5,f(1)],[1,0]),
  rectangle([1,f(1.5)],[1.5,0]),
  rectangle([1.5,f(2)],[2,0]),
  rectangle([2,f(2.5)],[2.5,0]),
  rectangle([2.5,f(3)],[3,0]),
  /* marcas eje-x */
  xtics = {["a=x0",-2],["x1",-1.5],["x2",-1],
          ["x3",-.5],["x4",0],["x5",0.5],["x6",1],
          [".....",2],["b=xn",3]},
  /* la curva */
  line_type=solid,
  line_width=3,
  explicit(f(x),x,-2,3),
     terminal = eps)$
```

7 Se reproduce aquí el examen de *Abitur*, junto con la solución parcial al mismo, con el amable permiso de su autor, el profesor Volker van Nek.

REFERENCIAS BIBLIOGRÁFICAS

ARI, N. (1989): *Application of computer code MACSYMA for electromagnetics*, en *Proceedings of the IASTED International Symposium*. Editor HAMZA, M. Expert Systems Theory and Applications, Anaheim, CA, USA, 1989. Acta Press.

BOGEN, R. (1986): *MACSYMA reference manual*. Symbolics, Inc. Cambridge Center, Cambridge, MA, USA, version 12 edition.

BRATCHER, K. (1989): *Utilization of the MACSYMA software for instructional programming*. Department of Mechanical Engineering, University of Louisville, Louisville, KY, USA.

CLARKSON, M. (1989): *An improved Laplace transform package for MACSYMA*. SIGSAM Bulletin, 19(2):31–33, Mayo.

DELEST, M. (1991): *Enumeration of polyominoes using Macsyma*. Theoret. Comput. Sci., 79(1):209–226, Febrero.

DELOATCH S. (1987): *MACSYMA usage at Langley*. NASA contractor report NASA-CR 172518, NASA, Washington, DC, USA.

FATEMAN, R. et al., editors (1977): *Proceedings of the 1977 MACSYMA Users' Conference*, number CP-2012 in NASA conference publication, Washington, DC, USA.

FEYNMAN, R. (2006) *¡Ojalá lo supiera! Las cartas de Richard Feynman*. Crítica, Barcelona.

GARCIA, A., MARTINEZ, A., MIÑANO, R. (1995): *Nuevas Tecnologías y Enseñanza de las Matemáticas*. Editorial Síntesis. Madrid.

HELLER, B. (1991): *MACSYMA for statisticians*. Wiley series in probability and mathematical statistics. Applied probability and statistics. John Wiley and Sons, New York, NY, USA.

IVIE, J. (1978): *Some Macsyma programs for solving recurrence relations*. ACM Transactions on Mathematical Software, 4(1):24–33, Marzo.

JONES, BARTLETT (1997): *Introduction to Macsyma*. Macsyma Inc. Sudbury, MA, USA.

KWOK, Y. (1991): *Application of MACSYMA to solutions of ordinary differential equations*. International journal of mathematical education in science and technology, 22(6), Noviembre.

MACSYMA Inc. (1995): *Macsyma user's guide*. Macsyma, Inc., Arlington, MA, USA, segunda edición.

MATHLAB GROUP (1983): *MACSYMA Reference Manual, Version Ten*. Massachusetts Institute of Technology, Computer Science Lab., Cambridge, MA, USA.

MAXIMA TEAM (2008a): *Maxima, a Computer Algebra System*.
http://maxima.sourceforge.net

MAXIMA TEAM (2008b): *Manual de Maxima. Traducción en español del Manual de Referencia*.
http://maxima.sourceforge.net/docs/manual/es/maxima.pdf

MAXIMA TEAM (2008c): *Documentation*.
http://maxima.sourceforge.net/documentation.html

SYMBOLICS Inc. (1988): *Macsyma User's Guide*. Burlington, MA, USA.

SYMBOLICS Inc. (1988): *MACSYMA reference manual*. Symbolics, Inc., Computer Aided Mathematics Group. Burlington, MA, USA, version 13 edition.

WOLFRAM, S. (1979) *MACSYMA tools for Feynman diagram calculations*, en *Proceedings of the 1979 MACSYMA Users Conference*, Editor LEWIS, E. Massachusetts Institute of Technology, Laboratory for Computer Science. Cambridge, MA, USA.