

Maxima con *wxMaxima*: software libre en el aula de matemáticas

J. Rafael Rodríguez Galván
Departamento de Matemáticas de la Universidad de Cádiz
Oficina de Software Libre de la Universidad de Cádiz

Febrero de 2007

Copyright © 2007, J. Rafael Rodríguez Galván. Versión 1.0.

Este documento es libre. Se otorga permiso para copiarlo, distribuirlo y/o modificarlo bajo los términos de la licencia FDL (GNU Free Documentation License) versión 1.2 o posterior, publicada por la Fundación de Software Libre¹. No contiene secciones invariantes, texto de portada ni de respaldo.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.

Las fuentes L^AT_EX pueden conseguirse en <http://knuth.uca.es/repos/maxima>

Los contenidos de la sección 1.3 están directamente basados en el manual “Primeros pasos en Maxima”, de Mario Rodríguez Riotorto. Copyright © 2006, Mario Rodríguez Riotorto.

¹<http://www.gnu.org/licenses/fdl.html>

Índice general

1. Generalidades	7
1.1. Introducción al software libre	7
1.1.1. Comunidades científicas de conocimiento y el origen del software libre	8
1.1.2. El software libre: definición y significado	9
1.1.3. El sistema operativo GNU/Linux	11
1.2. El software libre y las matemáticas en el aula	13
1.2.1. Software libre y educación	13
1.2.2. Software matemático	14
1.3. Breve historia de <i>Maxima</i>	17
2. Primer contacto con <i>Maxima</i> y <i>wxMaxima</i>	19
2.1. Instalación	19
2.1.1. <i>Maxima</i>	19
2.1.2. <i>wxMaxima</i>	20
2.2. Entornos de ejecución: la consola <i>Maxima</i> y <i>wxMaxima</i>	21
2.3. Primeros pasos y exploración del entorno <i>wxMaxima</i>	25
2.3.1. Abrir <i>Maxima</i> en modo consola	25
2.3.2. Primer contacto con <i>wxMaxima</i>	26
2.3.3. Sacar partido a las posibilidades de <i>wxMaxima</i>	30
2.3.4. Un ejemplo algo más avanzado	35
2.3.5. Una panorámica de <i>wxMaxima</i>	41
Referencias	45

Prólogo

El presente manual ha sido elaborado expresamente para el curso de formación al profesorado de los C.E.P. de Cádiz, Villamartín y Jerez que tendrá lugar en Febrero de 2007. La versión actual contiene una amplia guía de introducción a los programas *Maxima* y *wxMaxima* (situados en el contexto del software libre y las matemáticas) que deberá ser suficiente para utilizar con un cierto grado de pericia estas herramientas. A medida que avance el curso, se añadirán sucesivos capítulos que tendrán un carácter más específico, desarrollando completamente el programa de estos cursos. Los apuntes estarán disponibles en versión PDF a través del campus virtual del curso y en versión L^AT_EX en <http://knuth.uca.es/repos/maxima>.

1 Generalidades

Capítulo

1.1. Introducción al software libre

La mayor parte de los usuarios de informática están habituados a utilizar programas de ordenador con licencias privativas, sin ser conscientes de lo que esto significa. Es para ellos algo usual que el proceso de instalación de los programas incluya una etapa en la que se muestra la licencia y se pide al usuario que decida si la aceptan. Cosa que deben hacer si desean utilizarlo, aun siendo conscientes de que la aceptación de esta licencia puede significar la privación para realizar acciones que podrían ser tan normales como instalar el programa en un segundo ordenador, prestarlo o regalar una copia a otra persona. Es importante señalar que estas restricciones no son de tipo tecnológico (las nuevas tecnologías hacen que sea trivial la copia y difusión de información) sino que se basan en la elección de una licencia muy restrictiva por parte del autor y en la aceptación de ésta por parte del usuario.

Pero es más: si, teniendo suficientes conocimientos, detectamos un fallo en el programa y deseamos estudiarlo para poder solucionarlo por nosotros mismos, nos volveremos a encontrar con la prohibición legal de hacerlo, impuesta exclusivamente por la licencia que el autor eligió y que nosotros aceptamos. Por supuesto, no se puede trasladar esta situación a otras industrias diferentes a la del software. Los resultados serían paradójicos. Así, en la industria automovilística, la analogía sería la de una persona con suficientes conocimientos, por ejemplo un mecánico de profesión, que adquiere un coche. Años después el coche se para y éste decide abrir el capó para estudiar la avería y poder aplicar sus conocimientos para arreglarlo. Pero, descubre que esto es imposible pues, en el momento de la compra, el mecánico firmó un contrato que le prohíbe abrir el capó o realizar cualquier tipo de modificación técnica.

A este tipo de programas, cuyas licencias privan del derecho a estudiarlo, modificarlo o redistribuirlo, se les conoce como “software propietario” o, con más propiedad, “software privativo”.

Pero existen cada vez más programadores que, en ejercicio de sus legítimos derechos de propiedad intelectual, deciden dotar a su trabajo de una licencia distinta, mucho menos restrictiva, devolviéndole al propietario final sus derechos para copiarlo a cuantas personas desee, estudiarlo hasta el último detalle, y modificarlo sin restricciones para poder mejorarlo o adaptarlo. Los motivos para tomar esta decisión son muy variados: desde el mero altruismo hasta el más duro capitalismo, pasando por otras consideraciones, entre ellas políticas, empresariales o de simple eficacia.

Pero en última instancia, por muy novedoso que pueda parecer este concepto, es siempre conveniente tener en cuenta lo siguiente: cuando hablamos de software libre no

estamos haciendo más que trasladar al campo de la informática el método científico que, en los últimos siglos, ha dado excelentes resultados a la humanidad. En particular, para un matemático debería resultar natural la idea de aplicar a la informática (un área de conocimiento tan cercana) la forma en que las matemáticas se desarrollan y progresan.

1.1.1. Comunidades científicas de conocimiento y el origen del software libre

El modelo de una comunidad de personas que producen e intercambian información, criticándola y mejorándola para obtener, actualizar y publicar conocimiento, no es nuevo y tiene su paradigma en la comunidad científica y en las matemáticas.

En efecto, si bien en algún momento de la historia de las matemáticas éstas han podido inclinarse hacia el oscurantismo, apartándose del camino de la libre publicación y revisión por pares, el transcurrir de los siglos ha derivando inexorablemente hacia esta última senda. En la Europa del siglo XVII, como consecuencia de la revolución científica, se confirmó definitivamente este modelo. Y en el siglo XX, la irrupción de las tecnologías de la información y las comunicaciones, en particular de Internet, ha abierto posibilidades insospechadas para las matemáticas, la ciencia y en general para todas las comunidades basadas en el conocimiento, al facilitar la comunicación y la transmisión de información de forma instantánea a cualquier punto del planeta.

El concepto de software libre, tal y como lo conocemos hoy, se puede entender como el fruto de una concepción científica del conocimiento informático y de la generalización de las nuevas tecnologías y de Internet en las sociedades desarrolladas.

En la primera mitad del siglo XX y hasta la década de 1960, los ordenadores eran máquinas de grandes dimensiones instaladas en centros gubernamentales y grandes corporaciones y muchas de las personas encargadas de manipularlos formaban parte de la comunidad matemática y científica. En esta época, cuando una gran institución, pública o privada, adquiría un ordenador, el software venía como un acompañante y en este sentido se trataba como un todo con el hardware adquirido. Mientras se pagase el contrato de mantenimiento, se tenía acceso al catálogo de software que ofrecía el fabricante [2]. Además, no era común la idea de que los programas fueran algo separado desde un punto de vista comercial. En esta época, el software se solía distribuir junto a su código fuente y en general no había restricciones prácticas para usarlo, compartirlo o modificarlo. Podría decirse que el software, en su origen, fue libre y los informáticos constituían una comunidad científica que se podría comparar a la actual “comunidad de software libre”.

Pero en los años 70, las compañías de hardware comenzaron a vender parte de su software por separado. Con la percepción del software como un valor intrínseco, nació la preocupación por limitar (tanto técnica como legalmente) el acceso a los programas y restringir las posibilidades de los usuarios para compartirlo o modificarlo. Esto desembocó en la situación que todavía hoy es habitual, en la que el software con licencia privativa es mayoritario en el sector de la informática doméstica (aunque no en otros

sectores, como el de los servidores).

Aunque durante durante esta década siguieron apareciendo productos que hoy consideramos libres (como T_EX) o comunidades que compartían y desarrollaban el software de una forma abierta (por ejemplo, las comunidades que se formaron entorno al sistema operativo UNIX, que fueron semilla de los actuales sistemas libres de tipo BSD), no fue hasta principios de la década de 1980 cuando aparecieron, de forma organizada y consciente, los primeros proyectos para la creación de sistemas compuestos de software libre, y lo que probablemente es más importante: los fundamentos éticos, legales y hasta económicos, que luego se continuarían desarrollando hasta el día de hoy. De esta época procede también el propio término software libre, ambos formulados por primera vez por Richard Stallman a principios de los años 80.

Richard Stallman, empleado en el laboratorio de inteligencia artificial del prestigioso MIT (Instituto de Tecnología de Massachussets), se consideraba un *hacker*, en el sentido de una persona que disfruta ejercitando su inteligencia y compartiendo con sus colegas sus inquietudes tecnológicas y el código de sus programas. Pero en los primeros años 80 veía con desagrado cómo, progresivamente, sus compañeros de trabajo comenzaban a firmar contratos de exclusividad y no compartición de código para la elaboración de software privativo. Su negativa a hacerlo le estaban convirtiendo en un extraño en su propio mundo y, ante con la imparable extensión del software propietario en su entorno, se sentía limitado e impotente ante situaciones que antes podía solventar fácilmente.

En esta situación, tomó la decisión de abandonar su trabajo para poderse dedicar sin ataduras legales a un nuevo proyecto: la elaboración de un sistema de propósito general completamente libre [4], de tipo Unix, al que llamó GNU [1] (acrónimo recursivo que significa "GNU's Not Unix" es decir "GNU No es Unix").

1.1.2. El software libre: definición y significado

El software libre fue definido por R. Sallman como todo aquél que garantice las siguientes libertades:

0. Libertad para ejecutar el programa en cualquier lugar, en cualquier momento y con cualquier propósito.
1. Libertad de estudiar cómo funciona el programa, y adaptarlo a nuestras necesidades (requisito: acceso al código fuente).
2. Libertad para redistribuir copias a cualquier persona.
3. Libertad para mejorar el programa y publicar las mejoras (requisito: acceso al código fuente)

Esta definición es muy cercana a las características de las comunidades de conocimiento científico, en las que los avances se basan en la existencia de canales para el intercambio de conocimiento, la revisión por pares y la publicación de mejoras.

En la práctica, estas cuatro libertades se suelen concretar de acuerdo con la legalidad vigente por medio de una licencia, en la que se también se suelen plasmar algunas restricciones compatibles con ellas, como la obligación de dar crédito a los autores originales cuando redistribuyamos el trabajo. Algunos de los tipos de licencias libres más comunes:

- **BSD:** Estas licencias imponen muy pocas restricciones, estando muy cercanas al dominio público. Tanto es así que permiten incluso el que puedan existir derivaciones que no sean libres (este es, por ejemplo, el caso del sistema privativo Mac OS X, derivado de los BSD).
- **GPL:** Con el fin de proteger las cuatro libertades anteriores, se impone una restricción adicional, compatible con éstas: los trabajos derivados tienen que mantener la misma licencia libre que el trabajo original.

El mecanismo genérico que utilizan las licencias tipo GPL para conseguir estas garantías fue llamado *copyleft*, en un ingenioso juego de palabras, que hoy día es el nombre de una gran familia de licencias de software libre.

Es fundamental entender que con el concepto de software libre no estamos hablando simplemente de software gratuito: el software libre se puede vender si se desea (y en muchas ocasiones puede haber quien esté interesado en pagarlo). En este sentido, el software libre también puede ser “software comercial” y, de hecho, parte del modelo de negocio de algunas empresas (por ejemplo, distribuidoras de GNU/Linux como RedHat o SuSE) se centra en la venta de software libre. Aunque quien lo adquiera debe ser consciente de que (debido a la tercera libertad) podrá redistribuirlo cuando desee y como lo desee, por ejemplo sin pedir dinero a cambio ni permiso a nadie.

También es conveniente distinguir el software libre de otros conceptos, como *freeware* (software gratuito, pero sin libertad de estudio o modificación) o *shareware* (programas que se pueden evaluar de forma gratuita pero durante un tiempo, uso o características limitadas).

A nivel práctico, asociaremos el concepto de software libre con el de “software de fuente abierta” o de “código abierto” (*open source*), pues la única distinción es el enfoque que normalmente desean transmitir quienes utilizan esta denominación, mucho más pragmático, menos centrado en la ética y en la defensa de los derechos de los usuarios.

Existen decenas de miles de programas que pueden considerarse libres, desde pequeños juguetes de sólo unas líneas de código hasta sistemas operativos completos, como GNU/Linux, FreeBSD, etc, incluyendo a muchos de los programas más difundidos para servicios críticos en empresas e instituciones de todo el mundo. Algunos casos de éxito:

- **Mozilla Firefox.** Derivado del antiguo Netscape, Mozilla Firefox es el mejor navegador web existente en el mercado. Estable, seguro e integrado con los estándares de Internet, las características avanzadas que incorpora (uso de pestañas, filtros para ventanas emergentes, etc) marcan tendencia y, años más tarde, son incorporadas por el resto de los navegadores.

- *OpenOffice.org*, una completa *suite* ofimática (proceso de textos, hoja de cálculos, presentaciones...) que utiliza de forma nativa el estándar OpenDocument (ISO-26300) y es altamente compatible con MS Office.
- *Apache*, un servidor web con licencia libre que constituye el programa utilizado por el 70 % de los sitios web en el mundo, entre ellos servicios críticos de grandes corporaciones.
- En la lista que *Top500.org* publica semestralmente con los 500 supercomputadores más potentes del mundo. GNU/Linux ocupa el 73,40 % de los puestos (por ejemplo, Microsoft ocupa el 0,4 %, concretamente solamente solo hay dos ordenadores, que se sitúan en los lugares 132 y 472 del ranking).

1.1.3. El sistema operativo GNU/Linux

Aunque desde el principio el proyecto GNU, fundado en los años 80 para la creación de un sistema operativo completamente libre, incluyó en su sistema software que ya estaba disponible con una licencia que se podía considerar libre (como T_EX, o más adelante, el sistema X Window), había mucho que construir. Richard Stallman comenzó por escribir un editor (Emacs) y un compilador de C (GCC), ambos aún en uso (y muy populares) hoy día y más gente se unió a él.

A principios de la década de 1990, unos seis años después de su nacimiento, el proyecto GNU estaba muy cerca de tener un sistema completo similar a Unix. Sus productos ya gozaban de una merecida reputación de estabilidad y calidad y eran muy populares entre los usuarios de las distintas variantes de Unix, por aquella época el sistema operativo más usado en las empresas. Así, el proyecto GNU había conseguido ser relativamente conocido entre los profesionales informáticos y muy especialmente entre los que trabajaban en universidades. Aun así, hasta ese momento, todavía no había producido la pieza necesaria para culminar el rompecabezas: el núcleo del sistema, la pieza que se relaciona con el hardware y permite que todo funcione.

En julio de 1991 Linus Torvalds (por entonces, estudiante finés de 21 años) anunció por primera vez su proyecto de crear un núcleo de tipo Unix para sistemas Intel PC 386, al que llamó Linux. En septiembre libera la primera versión (0.01) y cada pocas semanas aparecen nuevas versiones. En marzo de 1994 apareció la primera versión que fue denominada estable (1.0), pero el núcleo desarrollado por Linus era usable desde bastantes meses antes. Durante este periodo, literalmente cientos de desarrolladores se habían volcado en su desarrollo y en su integración con el software de GNU y con muchos otros programas libres, como el sistema gráfico X-Windows, dando lugar al sistema operativo que hoy es conocido como GNU/Linux o, a veces, simplemente como Linux. Con el paso de los años este sistema operativo se extendió a arquitecturas distintas de Intel, como Sparc, ó AMD64.

Con GNU/Linux nacieron distintas *distribuciones*: selecciones de aplicaciones libres reunidas, configuradas y coordinadas que facilitan el proceso de instalación del sistema

para usuarios no expertos, a la vez que simplifican la gestión del software. Para ello, suelen utilizar paquetes de software, ficheros que contienen programas, documentación, etc. y que facilitan su instalación/desinstalación y configuración. Existen decenas de distribuciones, algunas con un carácter marcadamente comercial y otras manteniendo un espíritu más cercano al concepto de comunidad de intercambio de conocimiento. Entre las más conocidas: Debian, RedHat, Suse, Mandriva...

A finales de los años 1990, a la vez que otros sistemas operativos de escritorio (como Windows 95 y 98) se estaban extendiendo en los hogares, el sistema GNU/Linux continuaba creciendo y madurando. Entre sus bazas estaba el permitir convertir cualquier ordenador personal en una estación de trabajo de tipo Unix, segura, multiusuario, estable y orientada a Internet, lo que propició su implantación en universidades y en centros científicos, en ordenadores para cálculo numérico y en servidores, donde hoy goza de una posición privilegiada.

La aparición de sistemas de escritorio como GNOME y KDE, que forman una capa homogénea sobre X-Windows para interactuar con ventanas, menús, iconos, etc, ha permitido su acercamiento al gran público, así como la aparición de iniciativas políticas como la de la Junta de Extremadura y de Andalucía, que incentivan su uso en las escuelas y en la sociedad aprovechando, sus ventajas: ahorro de costes, impulso del desarrollo tecnológico local, reducción de la brecha digital frente a las grandes multinacionales del software, etc. En el mercado de los servidores, GNU/Linux es un "terreno neutral", atrayendo las inversiones de compañías de hardware (y software) de la talla de HP, Sun ó IBM.

En el caso de Andalucía, se optó por elaborar una distribución específica, a la que se llamó Guadalinux, que inicialmente estuvo basada directamente en Debian GNU/Linux y que ahora se basa en Ubuntu.

Debian es una distribución desarrollada a través de la colaboración de miles de voluntarios de todo el mundo. Se caracteriza por su fidelidad a la filosofía del software libre y por la diversidad de arquitecturas de ordenadores soportadas (entorno a 15 desde las derivadas del Intel 386 hasta las utilizadas en super-ordenadores). Pero, especialmente, por la calidad de su sistema de paquetes de software y por su abundancia (actualmente, en torno a 15.000).

Debido a estas características, suele tener el problema de su poca agilidad (poco frecuente actualización de sus versiones estables) y su dificultad de uso para los nuevos usuarios. Ubuntu es una distribución basada en Debian que, heredando muchas de sus ventajas, intenta aportar algunas nuevas, como facilidad de uso, introducción de las últimas versiones disponibles de los programas y ciclo regular de publicación de versiones del sistema (dos al año). Estas características motivaron que, a partir de la versión V3, Guadalinux utilizara a Ubuntu como base.

1.2. El software libre y las matemáticas en el aula

El uso de las tecnologías de la información y de la comunicación aumenta día a día y en todos los ámbitos. En particular, el uso del ordenador en las aulas de asignaturas de matemáticas está abriendo las puertas en nuestros días a nuevas oportunidades y a terrenos inexplorados. Nuestros alumnos constituyen una generación habituada a los contenidos audiovisuales y el uso de los ordenadores en el aula (obviamente sin ser la panacea que pueda resolver las dificultades del sistema educativo) puede constituir un recurso de tanta utilidad como la pizarra, el libro y el mapa.

1.2.1. Software libre y educación

La elección de los programas usados en el aula se debería guiar por una serie de consideraciones, de las cuales, calidad del producto y su adecuación a la asignatura son, por supuesto, requisitos previos. Pero existe una tercera cuestión que, en la práctica, tendrá una importancia crucial: la licencia de los productos elegidos y las connotaciones que este factor implica.

Una cuestión de precio La más evidente de estas consideraciones es simplemente una cuestión económica: para que un programa con licencia privativa pueda ser usado en las aulas, es necesaria la adquisición de suficientes licencias para cubrir los puestos de trabajo en las aulas de informática.

La copia ilegal La dependencia de programas privativos conlleva problemas éticos añadidos, puesto que de forma irremisible provoca en el alumnado y, en general, en toda la comunidad educativa, la *seducción por una marca cuyo precio hará que, en la mayoría de los casos, no pueda ser adquirida legalmente*, incitando su copia ilegal. Consciente de la actitud ilícita que provoca a sus alumnos, al profesor, que no puede ser guardián de los intereses de una empresa, le caben solamente dos opciones: o bien simular no ser consciente de esta situación, o bien apoyar abiertamente el que sus alumnos realicen actividades ilegales, como la copia de programas de ordenador cuyos autores no lo permiten.

Mejor con software libre En el otro extremo se sitúa el caso del software libre permitiendo que sea instalado en tantos ordenadores como sea conveniente y que el profesor comparta con sus alumnos, con toda legalidad, las herramientas utilizadas (quizás, acompañadas de material docente propio), facilitándoles reproducir en sus hogares el entorno de trabajo del aula. Más aún, al usar en el aula una herramienta con licencia libre, el profesor cuenta con ventajas adicionales a la hora de la planificación y el desarrollo de la asignatura, derivadas de *tener la garantía de que los programas podrán ser instalados y usado por los alumnos en su propio domicilio*, y además podrán ser instalados y usados por el profesor en tantos puestos como sea necesario.

Experimentar con varios programas El software con licencia libre permite y de hecho fomenta el *disponer de varias herramientas a la vez, complementarias o capaces de interactuar entre sí*, cada una de las cuales contará con sus puntos fuertes y sus debilidades. Aunque el profesor se decante por una de ellas, siempre podrá ofrecer a sus alumnos la enriquecedora posibilidad de experimentar con otras, de resolver un mismo problema desde distintas perspectivas y de saciar su curiosidad a aquellos que cuenten con mayores inquietudes.

Saciar la curiosidad Incluso, aunque sea muy poco el porcentaje de nuestros alumnos a los que esta cuestión les pueda interesar (de la misma manera que muy pocos de ellos se dedicarán a la física, a la literatura o al periodismo), el software libre ofrece la interesante posibilidad de *saciar la curiosidad de aquellos alumnos (¡y profesores!) interesados en saber cómo funcionan los programas de ordenador que están utilizando*. No solo esto, de *poder contribuir a mejorar los programas utilizados por sus compañeros o discípulos*. Y para ello no es necesario tener la categoría de gran experto: diseñar y enviar a los autores un nuevo icono, corregir o ampliar la documentación, proponer un ejemplo didáctico, sugerir un nuevo algoritmo... El poder observar y complementar un programa que es utilizado por miles de personas, profesionales y estudiantes de todo el mundo, constituye una experiencia tremendamente gratificante, de gran valor docente, como refuerzo y motivación.

Formación neutral Tengamos en cuenta que la velocidad con la que evoluciona la sociedad de la información puede hacer que tecnologías que hoy son hitos incuestionables sufran mañana severos cambios e incluso lleguen a ser superadas y olvidadas en cuestión de años: *¿qué procesador de textos estábamos usando hace una década?*. Por tanto *una formación basada en la excesiva dependencia de una única herramienta comercial, puede llegar, con el tiempo, a ser obsoleta*. Los estudiantes deberían estar formados en habilidades generales, en conocimiento neutral, y no en los productos concretos de una sola marca comercial. Sólo de esta manera se garantizará el carácter universal de los conocimientos adquiridos y se evitará que la no disponibilidad de un producto o sus carencias evidencien las lagunas del proceso formativo.

Valores éticos Y en último lugar, uno de los argumentos más importantes pero, con frecuencia, no suficientemente valorado, debido quizás al desconocimiento del software libre y a la asimilación social de los valores que conlleva el software privativo: *impulsando el software en el aula y con él los valores éticos asociados, estaremos basando la educación en pilares como la libertad, el conocimiento, la solidaridad y la colaboración*.

1.2.2. Software matemático

Para terminar este capítulo, introduciremos un listado de algunas de las aplicaciones con licencia libre que pueden resultar de utilidad para un profesor de matemáticas:

Geometría

- Dr. GEO (Geometría dinámica e interactiva). En Guadalinex
<http://www.offset.org/drgeo>
- Kig (Geometría interactiva). En Guadalinex
<http://edu.kde.org/kig/>
- Kseg (Geometría interactiva). En Guadalinex
<http://www.mit.edu/~ibaran/kseg.html>
- Eukleides (Lenguaje para construcción de figuras geométricas). En Guadalinex
<http://www.eukleides.org/>
- Geomview (Visor de objetos 3D interactivo). En Guadalinex
<http://www.geomview.org/>
- PyGeo (Marco para la creación de construcciones geométricas dinámicas, utilizando el lenguaje Python).
<http://pw1.netcom.com/~ajs/>

Aritmética

- Kcalcul (Operaciones aritméticas: suma, resta, multiplicación y división)
<http://website.lineone.net/~a-m.mahfouf/kcalcul.html>
- Kpercentage (Cálculo de porcentajes). En Guadalinex
<http://edu.kde.org/kpercentage/>
- Xabacus y Xmabacus (Operaciones con un ábaco). En Guadalinex
<http://ftp.tux.org/pub/tux/bagleyd/xabacus>

Representación gráfica

- GNUplot (Funciones y tablas de valores, 2d y 3d). En Guadalinex
<http://gnuplot.info/>
- Kmplot (Funciones en 2D). En Guadalinex
<http://edu.kde.org/kmplot/>
- Geg (Funciones en 2D). En Guadalinex
<http://www.infolaunch.com/~daveb/>

- Superficie (Superficies en 3d).

<http://superficie.sourceforge.net/>

Fractales

- Xaos (Visor interactivo de fractales en tiempo real). En Guadalinex

<http://xaos.sourceforge.net/>

Juegos

- Tux, of Math Command (tuxmath). (Dispara a las naves espaciales y resuelve operaciones matemáticas) En Guadalinex

<http://www.newbreedsoftware.com/tuxmath/>

- MathWar (Resolver operaciones matemáticas). En Guadalinex

<http://webpages.charter.net/stuffle/linux/software.html#mathwar>

Estadística

- R (excelente programa de cálculo estadístico). <http://www.r-project.org/>

Cálculo simbólico y numérico

- Octave (Excelente programa de cálculo matricial y numérico). En Guadalinex

<http://www.octave.org>

- Yacas (Cálculo simbólico). En Guadalinex

<http://yacas.sourceforge.net/>

- Axiom (Cálculo simbólico). En Guadalinex

<http://www.axiom-developer.org/>

- PARI/GP (Teoría de números). En Guadalinex

<http://www.parigp-home.de/>

- Y por supuesto... **Maxima** (Cálculo simbólico). En Guadalinex

<http://maxima.sourceforge.net/>

1.3. Breve historia de *Maxima*

Maxima es un programa cuyo objeto es la realización de cálculos matemáticos simbólicos (aunque también numéricos), capaz de manipular expresiones algebraicas, derivar e integrar funciones y realizar diversos tipos de gráficos.

Los orígenes de Maxima hay que buscarlos a partir del año 1967 en el MIT AI Lab (Laboratorio de Inteligencia Artificial del Instituto Tecnológico de Massachussets) como una parte del proyecto MAC (Machine Aided Cognition). El programa recibiría por aquel entonces el nombre de *Macsyima* (MAC's SYmbolic MAnipulator), del cual el MIT mandaría una copia en 1982 al DOE (US Department Of Energy), uno de los organismos que aportaron los fondos económicos para el desarrollo del proyecto; esta primera versión se la conoce como DOE-*Macsyima*. Posteriormente, el DOE concede la licencia de explotación del programa a la empresa Symbolics, que sigue desarrollando el proyecto durante unos años. En 1992 el programa es adquirido por una empresa que se llamaría precisamente *Macsyima Inc*, y el programa iría perdiendo fuelle progresivamente ante la presencia en el mercado de otros programas similares como *Maple* o *Mathematica*, ambos los dos inspirados en sus orígenes por el propio *Macsyima*.

Pero ocurrieron dos historias paralelas. Desde el año 1982, y hasta su fallecimiento en el 2001, William Schelter en la Universidad de Texas mantuvo una versión de este programa adaptada al estándar Common Lisp, la cual ya se conocía con el nombre de Maxima para diferenciarla de la versión comercial. En el año 1998 Schelter consiguió del DOE permiso para distribuir Maxima bajo la licencia GNU-GPL (<http://www.gnu.org/licenses/gpl.html>); con este paso, muchas más personas empezaron a dirigir su mirada hacia Maxima, justo en el momento en el que la versión comercial estaba prácticamente muerta. Actualmente, el proyecto es un programa escrito en lenguaje lisp que está siendo liderado por un grupo de desarrolladores originarios de varios países, asistidos y ayudados por otras muchas personas interesadas en Maxima y que mantienen un cauce de comunicación a través de una lista de correo (<http://maxima.sourceforge.net/maximalist.html>).

Puesto que Maxima se distribuye bajo la licencia GNU-GPL, tanto el código fuente como los manuales son de libre acceso a través de la página web del proyecto (<http://maxima.sourceforge.net>). *Maxima* es, por tanto, un potente motor de cálculo simbólico aunque, en su origen, no destacaba por tener una interfaz gráfica más amigable para los usuarios que la simple consola de texto. Con el tiempo este hecho ha ido cambiando y han aparecido distintos entornos de ejecución que intentan facilitar la interacción con los usuarios (y que se enumerarán en la sección 2.2). Entre ellos, este curso se centrará en *wxMaxima*, desarrollado por Andrej Vodopivec y disponible en <http://wxmaxima.sourceforge.net>.

2 Capítulo

Primer contacto con *Maxima* y *wx-Maxima*

2.1. Instalación

En este apartado se detallará el proceso e instalación de *Maxima* y *wxMaxima*, prestando especial atención al caso de Guadalinex (y, por tanto, de Ubuntu y sistemas derivados de Debian).

2.1.1. *Maxima*

Maxima puede funcionar en distintos sistemas operativos, entre ellos diversas variantes de Windows y de GNU/Linux. En el resto de esta sección nos centraremos en el caso de Guadalinex (V3 o superior aunque, en particular, todo lo que se comentará es aplicable de forma directa a Ubuntu o a cualquier otra distribución basada en Debian).

El lector interesado en utilizar *Maxima* en alguna otra variante de GNU/Linux o en Windows, puede acceder a la sección *Download* de la web de *Maxima* y seguir las instrucciones que en ella se indican.

Para utilizar este programa desde Guadalinex, podemos actuar de la forma habitual: abrimos un gestor de paquetes, por ejemplo *Synaptic* (menú "Sistema" → "Administración"), buscamos el paquete¹ "maxima", lo marcamos para instalar². Por último, aplicamos los cambios que hemos marcado³.

Además de "maxima", es recomendable instalar algunos otros paquetes complementarios:

- "maxima-share" Extensiones de *Maxima*, muchas de las cuales introducirán nuevas bibliotecas y funcionalidades que nos resultarán de gran utilidad.
- "maxima-doc" Documentación sobre *Maxima*, en particular el manual "oficial" del programa, que por defecto se instala en `/usr/share/doc/maxima/` y que puede ser abierto desde *wxMaxima*.
- "gnuplot-x11": representación de gráficos 2D y 3D.

¹Pinchando en el icono "Buscar", o bien el menú "Editar" → "Buscar..." o bien tecleando CTRL+F

²Ya sea haciendo doble *click* sobre él, o pulsando el botón derecho del ratón, o usando la entrada de menú "Paquete" → "Marcar para instalación" o bien pulsando CTRL+I

³Para ello, deberemos hacer *click* en el botón "Aplicar" o bien en el menú "Editar" → "Aplicar cambios marcados" o bien teclear CTRL+F

Debemos tener en cuenta que, puesto que no hemos instalado ninguna interfaz gráfica de usuario, para comenzar a utilizar *Maxima* será necesario abrir una consola (menú "Aplicaciones" → "Accesorios" → "Terminal") y teclear en ella el comando "maxima".

2.1.2. *wxMaxima*

La instalación de *wxMaxima* en Guadalinex, en Ubuntu o en cualquier sistema derivado de Debian, sigue el mismo proceso que se comentó en el apartado anterior:

- Abrir un gestor de paquetes, por ejemplo *Synaptic*
- Buscar el paquete "wxmaxima"
- Marcarlo para instalar
- Aplicar los cambios marcados

Una vez instalado en el sistema, contaremos con una nueva entrada en el menú (del tipo de "Aplicaciones" → "Otros" → "wxMaxima"), que es el camino que utilizaremos habitualmente para arrancar el programa.

Sin embargo, en Guadalinex V3 (no en V4 y posteriores) se nos presenta una dificultad: *wxMaxima* no se encuentra entre los paquetes disponibles, ya que en el momento del lanzamiento de su lanzamiento, el paquete *wxMaxima* aún no había sido incluido en Ubuntu, su distribución base.

La solución es simple⁴: se trata de crear un paquete de *wxMaxima* específicamente para Guadalinex V3. Esta es la tarea que se ha llevado a cabo en la Oficina de Software Libre de la Universidad de Cádiz, dando como resultado una serie de paquetes disponibles en <http://osl.uca.es/debian/breezy-backports/>.

Para instalar estos paquetes, es suficiente con añadir la dirección web anterior a la lista de repositorios de Guadalinex. Si usamos *Synaptic*, basta acceder a1 "Configuración" → "Repositorios" → "Añadir" → "Personalizado" , a continuación introducimos en "Línea de APT" lo siguiente:

```
deb http://softwarelibre.uca.es/debian breezy-backports/
```

A partir de ese momento, tendremos disponible el paquete "wxmaxima" (versión 0.7.0), que podrá ser instalado de la forma habitual. Al hacerlo, también será instalada la versión 5.10 de *Maxima*. Además, se podrá contar con la versión 5.10 del resto de los paquetes complementarios.

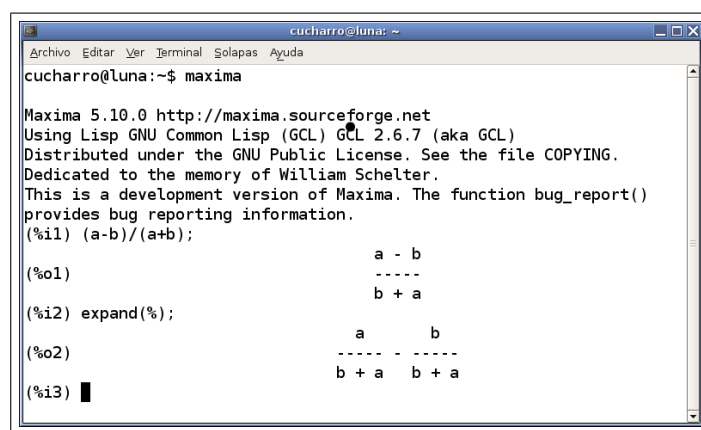
Para la instalación de *wxMaxima* en otras variantes de GNU/Linux o de Windows, se puede consultar wxmaxima.sourceforge.net. En el caso de Windows, resultará

⁴Aunque, técnicamente, para ponerla en práctica es necesario tener conocimientos sobre la creación de paquetes Debian, además ha sido necesario sortear algunos problemas de conexión entre *Maxima* y *wxMaxima* que han sido solucionados con la versión 5.10 del primero, lo que ha hecho necesario empaquetar también esta versión de *Maxima* y crear el resto de paquetes complementarios

necesaria una tarea adicional de post-instalación: indicar a *wxMaxima* cuál es la localización concreta del programa ejecutable de *Maxima* dentro del árbol de carpetas de Windows. Para ello, se debe utilizar el menú "Editar" → "Preferencias" y rellenar el campo "Programa Maxima:"

2.2. Entornos de ejecución: la consola *Maxima* y *wxMaxima*

Como se ha comentado anteriormente, *Maxima* es un potente motor de cálculo simbólico (y numérico). El paquete básico permite utilizar sus funcionalidades a través de una consola de texto (figura 2.1), preparada para que el usuario comience a introducir órdenes en el lenguaje de *Maxima*.



```
cucharro@luna:~$ maxima
Maxima 5.10.0 http://maxima.sourceforge.net
Using Lisp GNU Common Lisp (GCL) GCL 2.6.7 (aka GCL)
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) (a-b)/(a+b);
              a - b
              ----
              b + a
(%o1)
(%i2) expand(%);
              a      b
              ---- - ----
              b + a  b + a
(%o2)
(%i3) █
```

Figura 2.1: *Maxima* ejecutándose en entorno consola

Aunque puede resultar demasiado espartano para personas nuevas en *Maxima*, este entorno permite acceder a todas sus posibilidades y muchos de los usuarios más avanzados pueden preferir, en ocasiones, la claridad y velocidad que proporciona el acceso a las funcionalidades sin necesidad de navegar entre árboles de menús con el ratón.

Pero además, existen una serie de programas, entre ellos *wxMaxima*, que actúan como entornos gráficos, permitiendo al usuario el ejecutar *Maxima* de forma indirecta e interactuar con él mediante filosofías más "visuales" (figura fig:interfaz-wxmaxima). Estos entornos están dotados de licencia libre y se pueden instalar de forma complementaria a *Maxima*, siguiendo un proceso similar a lo que se comentó en el apartado anterior. Cada una de ellas tiene unas características propias que la pueden hacer más adecuada para unos usuarios u otros.

En este manual se ha optado por elegir desde el principio una de estas interfaces para fijar concretamente los contenidos y la forma de interactuar con el programa. La interfaz elegida es *wxMaxima*, debido a que se trata, quizás, de aquella que puede resultar más amable para un profesor o un estudiante que se enfrente por vez primera al uso de

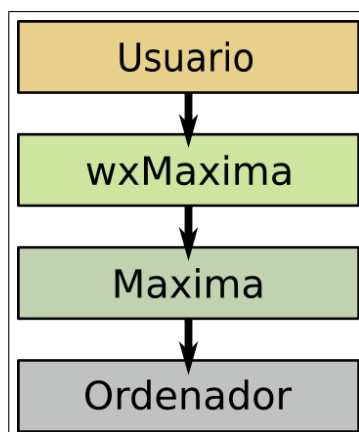


Figura 2.2: *wxMaxima* como interfaz de *Maxima*

Maxima dentro del aula de matemáticas. A partir de la próxima sección se comenzará a estudiar este programa con detalle.

Aun así, se ha creído conveniente ofrecer una breve panorámica de los entornos existentes, con la intención de ofrecer suficientes elementos de juicio como para que el lector interesado pueda explorar la riqueza que proporciona a *Maxima* el contar con tal diversidad de interfaces:

- *xmaxima* (figura 2.3), la primera interfaz gráfica que fue desarrollada, es mantenida “oficialmente” por el equipo de desarrollo de *Maxima*. De hecho, en algunos sistemas como Windows, se instala automáticamente y es arrancada por defecto. Presenta algunas ventajas, como la integración, en formato HTML, de manuales de ayuda. Sin embargo, también tiene algunas desventajas con respecto a otras interfaces más modernas como *wxMaxima* y presenta menos ventanas y menús de apoyo que ésta.
- *TEXmacs* (figura 2.4) es un proyecto iniciado en 1998 el C.N.R.S (Instituto Nacional Francés para la Investigación Científica), con el propósito de crear una aplicación para redactar textos matemáticos de forma sencilla. Su nombre tiene raíces en el sistema $\text{T}_{\text{E}}\text{X}$, en el que se basa, y en el conocido editor de textos Emacs, del que toma parte de su filosofía.

El resultado es un editor de textos de tipo WYSWYG⁵ que permite la creación de documentos matemáticos y científicos con la calidad que aporta $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$, utilizado como motor de maquetado.

Pero el verdadero potencial de *TEXmacs* estriba en la posibilidad de incorporar y utilizar con comodidad sesiones interactivas de numerosos motores de cálculo, entre ellos *Maxima*. En este sentido, *TEXmacs* es una “interfaz universal” para programas matemáticos, aportando a los mismos ventajas adicionales, como un modo

⁵What You See is What You Get, lo que ves (en pantalla) es lo que obtienes (al imprimir)

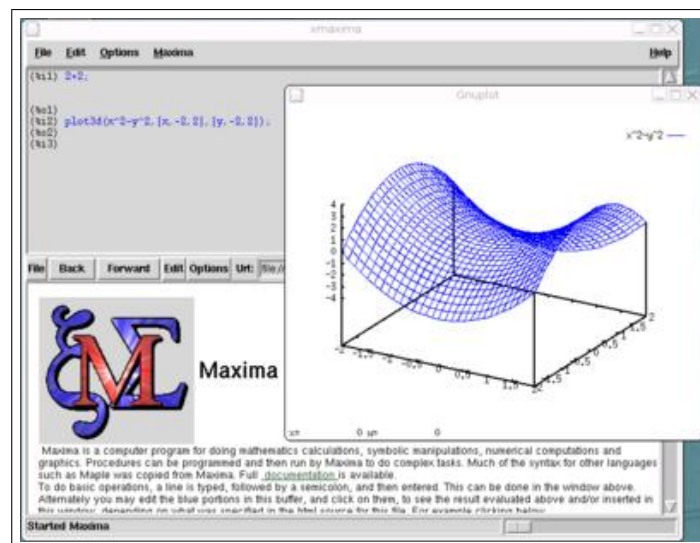


Figura 2.3: *Maxima* ejecutándose bajo la interfaz *wxMaxima*

(algo limitado) para la introducción de expresiones con el ratón y, en especial la mejora del aspecto de las salidas matemáticas.

Pero, por otra parte, el carácter genérico de *T_EXmacs* hace que no incluya acceso mediante menús o ventanas a las funciones específicas de *Maxima*.

- *Emacs* (figura 2.5) es un editor de textos completamente configurable que constituye una plataforma de desarrollo con modos específicos para decenas de lenguajes de programación (entre ellos, el lenguaje de *Maxima*) y con soporte para la edición de ficheros *T_EX/L_AT_EX*.

Aunque cuente con un modo que permite ejecutar una sesión *Maxima* (para ello, debemos teclear M-X y escribir “maxima”), los usuarios más avanzados podrán encontrar más interesante la posibilidad de editar ficheros en lenguaje *Maxima* (con reconocimiento y coloreado de sintaxis y aprovechando las ventajas de un editor tan avanzado). Estos ficheros pueden ser enviados a *Maxima* para su evaluación, completamente o línea a línea (para ello, en *wxMaxima* se puede utilizar el menú “Archivo” → “Archivo de lotes”).

Por último, un par de modos que pueden resultar interesantes a la hora de usar *Maxima* en *Emacs*: Por un lado, “imaxima”, que se puede utilizar para que *Emacs* use *T_EX* para embellecer las salidas de *Maxima* (como se puede ver en la ventana derecha de la figura 2.5). Y por otro lado, “emaxima”, ideado para la redacción de documentos *T_EX/L_AT_EX* en los que haya que incluir sesiones o comandos de *Maxima*. Este es el modo se ha utilizado para la redacción del presente manual.

- ... y, por supuesto, *wxMaxima*, basada en la biblioteca gráfica *wxwidgets*, gracias a la cual existen versiones nativas tanto para sistemas operativos GNU/Linux co-

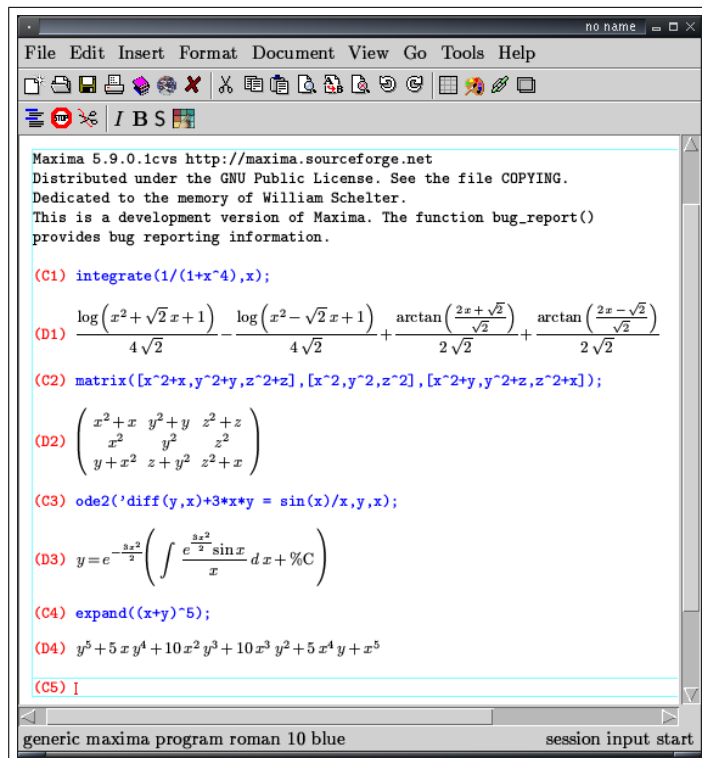


Figura 2.4: *Maxima* ejecutándose en una sesión $T_{E}X$ macs

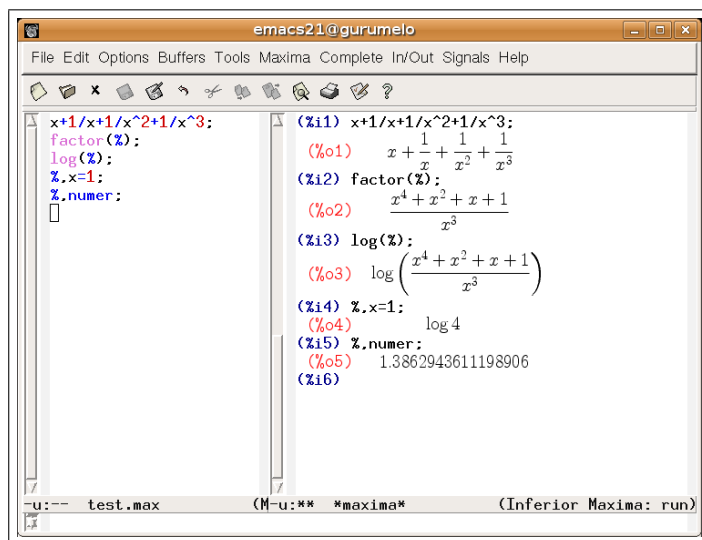


Figura 2.5: Un fichero maxima (*buffer* izquierdo) y una sesión imaxima (*buffer* derecho). Ambos, interactuando en *Emacs*

mo para Windows. Integra elementos específicos para la navegación de la ayuda, introducción de matrices, creación de gráficas, cálculo de límites, derivadas o integrales, etc. A partir de la próxima sección se estudiará con más detalle.

2.3. Primeros pasos y exploración del entorno *wxMaxima*

2.3.1. Abrir *Maxima* en modo consola

La experiencia de un primer contacto con *Maxima* puede resultar completamente diferente según la interfaz que empleemos.

En el caso de utilizar una consola de texto (y, en mayor medida, en otras interfaces, como *xmaxima* o *TEXmacs*), contaremos básicamente con una pantalla de bienvenida similar a la siguiente:

```
Maxima 5.10.0 http://maxima.sourceforge.net
Using Lisp GNU Common Lisp (GCL) GCL 2.6.7 (aka GCL)
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)
```

A través de estas líneas, *Maxima* nos ofrece diversos datos, entre ellos la versión con la que estamos trabajando y la dirección web del proyecto. La clave está en el indicador que aparece en la última línea, con la etiqueta `(%i1)` seguida del cursor en indicación de que se encuentra esperando a la primera entrada (en inglés *input*, de donde proviene la letra “i” de la etiqueta) para interactuar con el programa.

Podemos, por ejemplo, calcular una suma muy sencilla, tecleando la operación deseada seguida de un punto y coma (“;”) y una pulsación de la tecla RETORNO. *Maxima* procesará la operación y nos devolverá el resultado, precedido de la etiqueta `(%o1)` (del inglés *output* 1). Por ejemplo:

```
(%i1) 2+2;
(%o1) 4
(%i2)
```

La etiqueta `(%i2)` irá seguida del cursor, en espera de una segunda instrucción.

A partir de este momento, podremos acceder a todas las posibilidades que encierra *Maxima*, eso sí, siempre que contemos con un manual adecuado. En este sentido, se puede recomendar el de Mario Rodríguez Riotorto (Primeros Pasos en Maxima) [3] y, por supuesto, el manual “oficial” [5].

2.3.2. Primer contacto con *wxMaxima*

Por otro lado, cuando accedemos a *Maxima* a través de *wxMaxima* (pinchando en el menú de aplicaciones de nuestro entorno de ventanas), encontraremos una ventana amigable, con numerosos botones y menús. La podemos considerar dividida en distintas secciones (ver la figura 2.6):

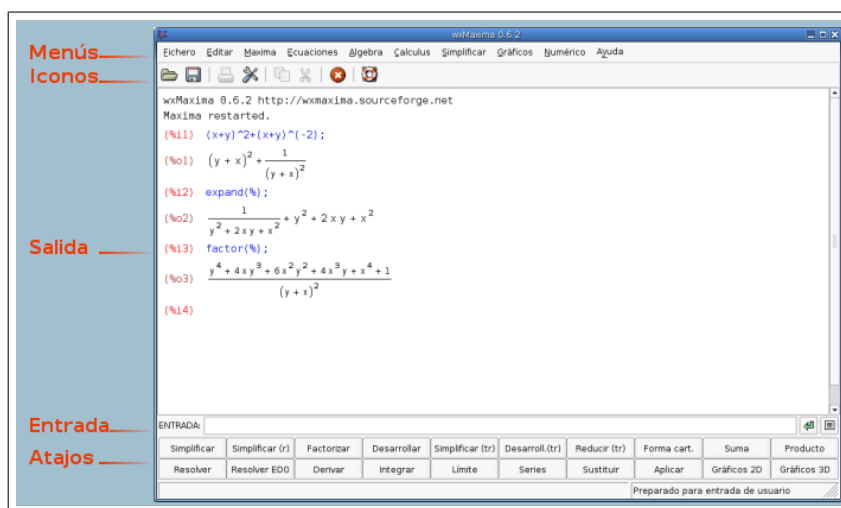


Figura 2.6: Distintas áreas de *wxMaxima*

1. *Barra de menús*: Nos permite acceder al motor de cálculo simbólico *Maxima*
2. *Barra de iconos*: Acceso rápido a algunas de las opciones de la barra de menús
3. *Área de salida o consola*: En ella se muestran los resultados
4. *Área de entrada*: Para teclear comandos
5. *Área de botones o atajos*: Otro punto de acceso rápido a los comandos de *Maxima*

Inicialmente, el *área de salida* contiene una información similar a la que antes apreciábamos, con información sobre el programa:

```
wxMaxima 0.6.4 http://wxmaxima.sourceforge.net
Maxima restarted.
(%i1)
```

Como antes, comenzaremos realizando una simple suma. Para ello, en esta ocasión nos debemos situar en el *área de entrada* (con el ratón o pulsando la tecla F4) y teclear la operación que deseemos, por ejemplo, "44+77" (ver figura 2.7). A continuación pulsamos la tecla de retorno.

ENTRADA:

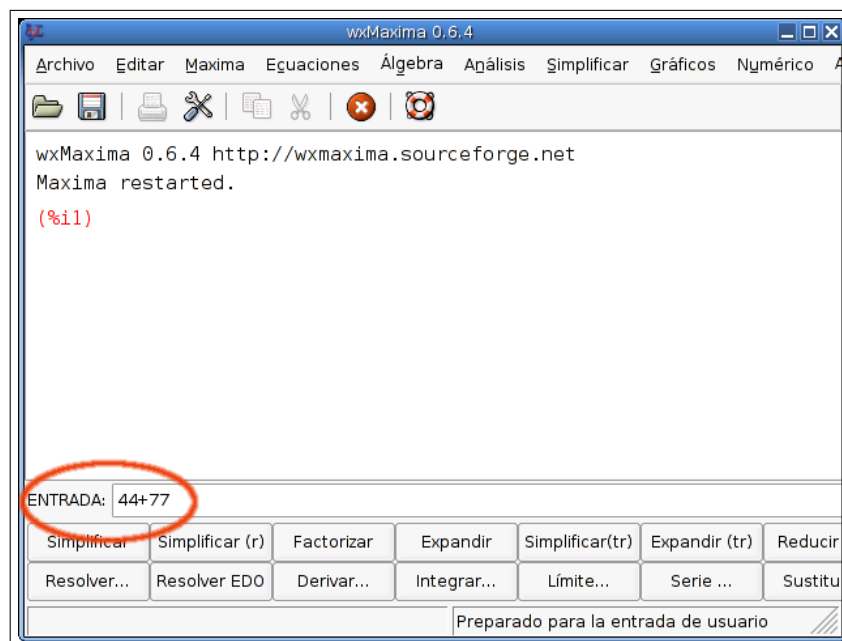


Figura 2.7: Utilización del área de entrada

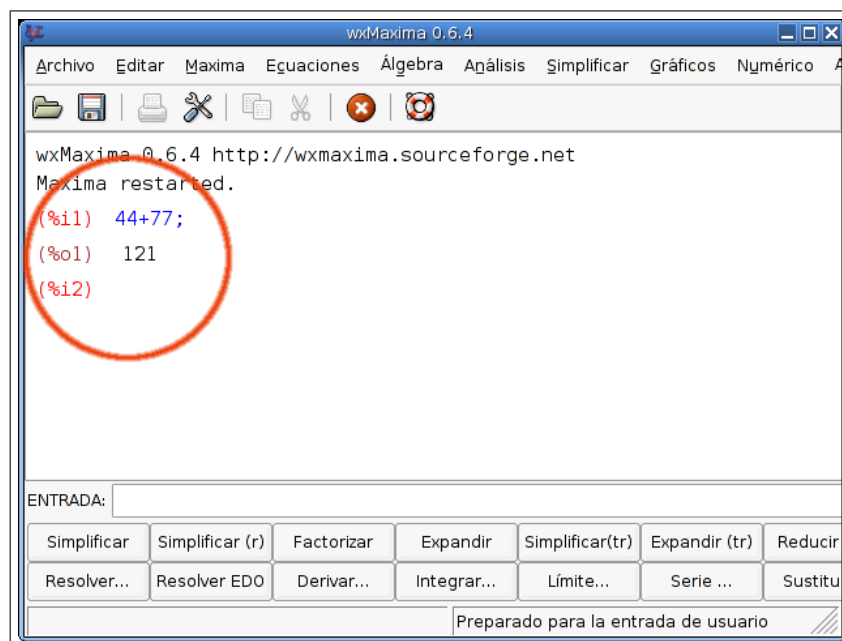


Figura 2.8: Resultados obtenidos en el área de salida

wxMaxima responderá mostrando, en el área de consola, el resultado de la operación anterior (ver figura 2.8).

En el diseño del área de entrada se han incluido algunas características destinadas a facilitar la introducción de expresiones del lenguaje *Maxima*. Por ejemplo, ya no es necesario que las entradas utilicen el carácter punto y coma para indicar el fin de línea, como se puede apreciar en el ejemplo anterior. Pero el punto y coma en *Maxima* actúa también como un separador cuando escribimos varias instrucciones seguidas. Nuestro siguiente ejemplo consistirá en asignar el valor 32123 a la variable x , el 1234321 a y , para luego solicitar su producto. Para ello, tecleamos:

ENTRADA:

y obtenemos una salida similar a la siguiente:

```
(%i1)  x:32123;
(%o1)
                                     32123

(%i2)  y:1234321;
(%o2)
                                     1234321

(%i3)  x*y;
(%o3)
                                     39650093483
```

Como se puede apreciar, en el lenguaje de *Maxima* se utiliza el símbolo de dos puntos (":") para asignar un valor a una variable. El símbolo de igualdad ("=") queda reservado para las ecuaciones.

En ocasiones, no nos interesará que aparezcan en pantalla los valores de algunas operaciones, por ejemplo la asignación de valores a variables intermedias, para lo cual podemos utilizar el carácter "\$" como sustituto del separador ";". Por ejemplo, si escribimos:

ENTRADA:

obtenemos una salida similar a la siguiente, mucho más limpia que en ejemplo anterior y similar a la siguiente:

```
(%i4)  x:321123$
(%i5)  y:123321$
(%i6)  x/y;
(%o6)
                                     263
                                     101
```

Como vemos, *Maxima* opera con aritmética racional y, por defecto, nos devuelve una fracción como resultado. Si añadimos una coma (",") seguida de la orden "numer", se obtendrá una expresión numérica, por defecto, con 16 cifras decimales. Por ejemplo, realizaremos la siguiente resta:

ENTRADA:

(%i7) 5-2/3;

(%o7)

$$\frac{13}{3}$$

y pediremos cuál es el valor numérico de la salida anterior:

ENTRADA:

(%i8) %, numer;

(%o8)

4,3333333333333333

En este ejemplo hemos utilizado por primera vez el operador "%", que se emplea para hacer referencia a la última salida. Se pueden usar las etiquetas "%iN" y "%oN" para acceder, respectivamente, al valor de la entrada y la salida N-ésima (por ejemplo, "%o3" es una referencia a la salida número 3).

Por último, *Maxima* puede trabajar, no solamente con números y variables, sino también con expresiones simbólicas. Por ejemplo, escribimos:

ENTRADA:

(%i9) a+b+a/b;

(%o9)

$$b + \frac{a}{b} + a$$

Y a continuación usamos la función de *Maxima* "ratsimp", que simplifica expresiones racionales, expresándolas de una forma canónica.

ENTRADA:

(%i10) ratsimp(%);

(%o10)

$$\frac{b^2 + ab + a}{b}$$

En este caso, debemos ser cuidadosos, pues si las variables utilizadas tienen algún valor, obtendremos un resultado numérico. Por ejemplo, puesto que en un ejemplo anterior habíamos definido x e y , tendríamos:

ENTRADA: `x+y+x/y`

(%i11) `x+y+x/y;`

(%o11)

$$\frac{44889107}{101}$$

Para poder emplear estas variables de forma simbólica tendremos que eliminar su valor:

ENTRADA: `kill(x,y)`

(%i12) `kill(x,y);`

(%o12)

done

ENTRADA: `x+y+x/y`

(%i13) `x+y+x/y;`

(%o13)

$$y + \frac{x}{y} + x$$

En *wxMaxima*, la entrada "`kill(x,y)`" se puede introducir mediante la entrada de menú "*Maxima*" → "*Borrar Variable*". Y existen muchas otras ventajas similares que se comenzarán a introducir a partir del siguiente apartado.

2.3.3. Sacar partido a las posibilidades de *wxMaxima*

Hasta este momento, nos hemos limitado a utilizar *wxMaxima* como un marco para introducir las órdenes de *Maxima*, sin utilizar más elementos que las áreas de entrada y salida. Pero el entorno gráfico de *wxMaxima* tiene muchas más funcionalidades, algunas de las cuales se mostrarán en los siguientes párrafos.

Por ejemplo, planteemos la siguiente relación de problemas (escogidos sin mayor finalidad didáctica que el ilustrar el funcionamiento de *wxMaxima*):

Problema:

- (a) Descomponer el valor de $10!$ en producto de factores primos
- (b) Factorizar el polinomio $x^6 - 1$
- (c) Multiplicar los factores obtenidos y comprobar que el resultado coincide con el polinomio anterior
- (d) Simplificar la siguiente fracción algebraica:

$$\frac{x^6 - 1}{x^2 + x + 1}$$

- (e) Resolver la ecuación

$$\frac{x^6 - 1}{x^2 + x + 1} = 0$$

Una persona con suficientes conocimientos del lenguaje de *Maxima* los podría haber resuelto mediante la siguiente secuencia de instrucciones:

```
(%i14) factor(10!);
(%o14)
```

$$2^8 3^4 5^2 7$$

```
(%i15) factor(x^6 - 1);
(%o15)
```

$$(x - 1) (x + 1) (x^2 - x + 1) (x^2 + x + 1)$$

```
(%i16) expand(%);
(%o16)
```

$$x^6 - 1$$

```
(%i17) ratsimp((x^6-1)/(x^2+x+1));
(%o17)
```

$$x^4 - x^3 + x - 1$$

```
(%i18) solve(%=0);
(%o18)
```

$$\left[x = 1, x = -1, x = -\frac{\sqrt{3}i - 1}{2}, x = \frac{\sqrt{3}i + 1}{2} \right]$$

La hipotética persona que resolvió el ejercicio anterior habría podido utilizar cualquiera de los entornos de ejecución de *Maxima* que se estudiaron en la sección 2.2. Entre ellos, podría haber optado por usar la línea de entrada de *wxMaxima* para introducir cada una

de las órdenes anteriores y conseguir, de esa manera, factorizar números y polinomios, simplificar fracciones, resolver ecuaciones, etc. Pero sacar realmente partido del entorno *wxMaxima* significa saber utilizar el gran número de utilidades, menús, botones y ventanas específicas que este programa pone a nuestra disposición para facilitarnos el trabajo. La idea es conseguir que, mediante el uso del ratón, los nuevos usuarios puedan realizar un acercamiento a *Maxima* tan amable como sea posible⁶ y los usuarios avanzados puedan aprovechar aquellas características que le puedan ser de utilidad.

De esta forma, aunque no conociéramos las funciones “`factor()`”, “`expand()`”, “`ratsimp()`”, “`solve()`” que fueron utilizadas anteriormente para resolver el problema, podríamos utilizar *wxMaxima* y actuar de la siguiente forma:

- (a) Introducimos el número 10!:

ENTRADA:

(%i19) 10!;

(%o19)

3628800

Como podemos ver, *wxMaxima* nos devuelve el valor del factorial.

- (b) Pulsamos el botón [Factorizar] situado en la barra de botones inferior (o utilizamos el menú “Simplificar” → “Factorizar expresión”). *Maxima* nos devuelve el siguiente resultado,

(%i20) factor(%);

(%o20)

$2^8 3^4 5^2 7$

- (c) Para factorizar el polinomio $x^6 - 1$ procedemos de la misma forma:

- a) Introducimos el polinomio:

ENTRADA:

- b) Pulsamos el botón [Factorizar] (o utilizamos el menú “Simplificar” → “Factorizar expresión”)

⁶Aunque, a medida que aumente su experiencia, puede que muchos usuarios encuentren más eficiente el conocer en profundidad las expresiones en el lenguaje *Maxima*, y utilizar los menús solamente en los casos en los que les resulten realmente necesarios.

Obtendremos el siguiente resultado en el área de consola:

```
(%i21) x^6-1;
```

```
(%o21)
```

$$x^6 - 1$$

```
(%i22) factor(%);
```

```
(%o22)
```

$$(x - 1) (x + 1) (x^2 - x + 1) (x^2 + x + 1)$$

- (d) Una vez factorizado, podemos volver a desarrollar el polinomio sin más que pulsar el botón [Expandir] (o el menú "Simplificar" → "Expandir expresión"), obteniendo:

```
(%i23) expand(%);
```

```
(%o23)
```

$$x^6 - 1$$

- (e) Para simplificar la fracción

$$\frac{x^6 - 1}{x^2 + x + 1}$$

podemos actuar de forma análoga: tecleamos

ENTRADA:

```
(%i24) (x^6-1)/(x^2+x+1);
```

```
(%o24)
```

$$\frac{x^6 - 1}{x^2 + x + 1}$$

y pulsamos el botón [Simplificar]

```
(%i25) ratsimp(%);
```

```
(%o25)
```

$$x^4 - x^3 + x - 1$$

(f) A continuación, podemos resolver la ecuación

$$\frac{x^6 - 1}{x^2 + x + 1} = 0$$

o, equivalentemente,

$$x^4 - x^3 + x - 1 = 0.$$

Para ello, no tenemos más que pulsar el botón [Resolver...] (o el menú "Ecuaciones" → "Resolver..."), tras lo cual se abrirá una ventana de diálogo (ventana *Resolver*, figura 2.9) en la que se nos preguntará cuáles son la ecuación y las variables para las que queremos resolver.



Figura 2.9: Ventana de diálogo *Resolver*

Por defecto, estará seleccionada la expresión anterior⁷ (símbolo %) para la variable x . Si estamos de acuerdo, no tenemos más que pulsar en el botón [Aceptar] para obtener el resultado:

```
(%i26) solve([%], [x]);
```

```
(%o26)
```

$$\left[x = 1, x = -1, x = -\frac{\sqrt{3}i - 1}{2}, x = \frac{\sqrt{3}i + 1}{2} \right]$$

Algunas observaciones:

- En el ejemplo anterior, al utilizar el botón [Resolver], la expresión que estamos resolviendo (representada por la variable %) no es una ecuación, sino un polinomio ($x^4 - x^3 + x - 1$). En tal caso, *Maxima* trata de resolver la ecuación homogénea ($x^4 - x^3 + x - 1 = 0$).

⁷Aunque si hubiéramos seleccionado con el ratón alguna expresión del área de salida o si la hubiéramos tecleado en la línea de entrada, sería esta la que aparecería automáticamente como ecuación a resolver.

- Como se puede apreciar, *wxMaxima* introduce entre corchetes (“ [· · ·] ”) tanto las ecuaciones como las variables y las soluciones. Los corchetes representan listas de datos y es la estructura sobre la que descansa el lenguaje de programación Lisp y, por tanto, *Maxima*. Se volverá a hablar de ellos cuando estudiemos características avanzadas de este programa.
- Como veremos, *Maxima* utiliza listas de ecuaciones y de variables para resolver sistemas de ecuaciones lineales. En *wxMaxima* se puede utilizar para ello el menú “Ecuaciones” → “Resolver sistema lineal...” .

Problema:

Como ejercicio, se propone resolver el siguiente sistema de ecuaciones:

$$\left. \begin{array}{l} x + y = 0 \\ 2x + 3y = -1 \end{array} \right\}$$

2.3.4. Un ejemplo algo más avanzado

Planteemos ahora un problema que nos puede servir como un modelo más completo, relacionado con la representación gráfica de funciones.

Problema:

Dada la función

$$f(x) = \frac{\sqrt{x}}{1 + x^2}$$

- Estudiar su dominio, puntos de corte y asíntotas.
- Calcular su función derivada primera. ¿Está dicha función definida en $x = 0$? Hallar la derivada en el punto $x = 1$.
- Determinar sus intervalos de crecimiento y decrecimiento, así como sus máximos y mínimos relativos.
- Representar su gráfica.

Por supuesto, *Maxima* tiene instrumentos para la representaciones gráficas 2D y 3D pero, antes de utilizarlas, nos interesaremos por realizar un estudio analítico con la simple intención de mostrar la forma de proceder con *wxMaxima*:

- Al estudiar el dominio, la raíz cuadrada nos indica que debe ser $x \geq 0$. Pero también deberíamos plantearnos si es posible que el denominador sea cero. Por supuesto, esto es imposible, pues $1 + x^2$ siempre toma valores estrictamente positivos. Pero si hubiera alguna duda, se podría utilizar *Maxima*, para comprobar que $1 + x^2$ no tiene ninguna raíz real (todas sus soluciones son números imaginarios puros). Escribimos la ecuación (obsérvese que utilizamos para ello el símbolo “=”):

ENTRADA:

y a continuación pulsamos el botón [Resolver]

```
(%i1) 1+x^2=0;
```

```
(%o1)
```

$$x^2 + 1 = 0$$

```
(%i2) solve(%);
```

```
(%o2)
```

$$[x = -i, x = i]$$

Un inciso: si tuviéramos más conocimientos de *Maxima*, podríamos haber utilizado la orden "is" (que intenta comprobar si una expresión es cierta). Así, tecleamos lo siguiente (si deseamos escribir lo menos posible podemos copiar y pegar o bien escribir "is(%o4)"):

ENTRADA:

y obtenemos la expresión "false" (falso):

```
(%i3) is(1+x^2=0);
```

```
(%o3)
```

false

Podemos incluso concretar más: si preguntáramos

ENTRADA:

obtendríamos la respuesta "true" (cierto):

```
(%i4) is(1+x^2>0);
```

```
(%o4)
```

true

Continuando con el problema, vamos a estudiar los puntos de corte. Para ello, definiremos la función $f(x)$, utilizando para ello el operador "==" de la siguiente forma:

ENTRADA:

```
(%i5) f(x):=sqrt(x)/(1+x^2);
```

```
(%o5)
```

$$f(x) := \frac{\sqrt{x}}{1+x^2}$$

Se puede observar que hemos empleado la función “sqrt” para la raíz cuadrada. A continuación, podemos estudiar el punto de corte con el eje de ordenadas, obteniendo el punto (0, 0):

ENTRADA:

```
(%i6) f(0);
```

```
(%o6)
```

0

Para obtener los puntos de corte con el eje de abscisas, podemos pulsar el botón [Resolver] y escribir la ecuación “f(x)=0” (o simplemente “f(x)”), obteniendo de nuevo el punto $x = 0$:

```
(%i7) solve([f(x)=0],[x]);
```

```
(%o7)
```

[x = 0]



Figura 2.10: Ventana de diálogo *Límite*

Para estudiar las asíntotas horizontales (no existen verticales), podemos utilizar el botón [Límite...], que abrirá una ventana de diálogo (figura 2.10) en la que teclearemos la expresión “f(x)” y el valor al que tiende la variable x . Si lo

deseamos, podemos utilizar el botón [Especial] (contenido en la ventana *Límite*) para acceder a los valores de $\pm\infty$ ("Infinity" y "- Infinity"). Obtenemos así:

```
(%i8) limit(f(x), x, inf);
```

```
(%o8)
```

0

```
(%i9) limit(f(x), x, minf);
```

```
(%o9)
```

0

- (b) *wxMaxima* nos pone fácil el cálculo de la derivada, a través del menú "Análisis" → "Derivar..." . Si tenemos el panel de botones completo (menú "Editar" → "Preferencias") tendremos también disponible el botón [Derivar...]. Se abrirá una ventana de diálogo (figura 2.11) en la que introduciremos la expresión a derivar ("f(x)"), la variable (x) y el orden de la derivada (1). El resultado es:



Figura 2.11: Ventana de diálogo *Derivar*

```
(%i10) diff(f(x), x);
```

```
(%o10)
```

$$\frac{1}{2\sqrt{x}(x^2+1)} - \frac{2x^{\frac{3}{2}}}{(x^2+1)^2}$$

expresión que queda más compacta si sumamos las dos fracciones, por ejemplo pulsado el botón [Factorizar]:

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

```
(%o11)
```

$$-\frac{3x^2-1}{2\sqrt{x}(x^2+1)^2}$$

En el cociente anterior se aprecia perfectamente que la función no está definida en $x = 0$, pues en tal caso estaríamos dividiendo por cero. Esto se ve con claridad si sustituimos $x = 0$ de la siguiente forma (también podíamos haber utilizado el menú "Simplificar" → "Sustituir..."):

ENTRADA:

El programa nos advierte de una división por cero (*Division by 0*).

Para hallar la derivada en el punto $x = 1$, podemos seleccionar con el ratón la expresión de la derivada y pulsar en el menú "Simplificar" → "Sustituir..." , sustituyendo x por 1 y obteniendo:

```
(%i12) subst(1, x, -(3*x^2-1)/(2*sqrt(x)*(x^2+1)^2));
(%o12)

$$-\frac{1}{4}$$

```

Otra posibilidad habría sido escribir simplemente

ENTRADA:

suponiendo que (%o16) sea la etiqueta correspondiente a la expresión de la derivada

- (c) Para estudiar el crecimiento y extremos relativos de $f(x)$ estudiaremos sus puntos críticos. Comenzamos definiendo una función $df(x) := f'(x)$ (lo que nos resultará más fácil si seleccionamos la expresión de la derivada (pulsamos "Editar" → "Selección a la entrada" o utilizamos cortar y pegar o la etiqueta %oN con N adecuado).

```
(%i13) df(x) := -(3*x^2-1)/(2*sqrt(x)*(x^2+1)^2);
(%o13)
```

$$df(x) := \frac{-(3x^2 - 1)}{2\sqrt{x}(x^2 + 1)^2}$$

Usando el botón [Resolver], obtenemos

```
(%i14) solve([df(x)=0], [x]);
(%o14)
```

$$\left[x = -\frac{1}{\sqrt{3}}, x = \frac{1}{\sqrt{3}} \right]$$

El primero de estos puntos críticos no está en el dominio de $f(x)$. Ahora podemos dar valores a $f'(x)$ en distintos intervalos, por ejemplo:

```
(%i15) 1/sqrt(3), numer;
```

```
(%o15)
```

0,57735026918962584

```
(%i16) df(0.1);
```

```
(%o16)
```

1,5034846242345494

```
(%i17) df(1);
```

```
(%o17)
```

$-\frac{1}{4}$

Así, $f'(x) > 0$ en $(0, 1/\sqrt{3})$ (f creciente) y $f'(x) < 0$ en $(1/\sqrt{3}, +\infty)$ (f decreciente). Por supuesto, tenemos un máximo relativo (y, en este caso, global) en $x = 1/\sqrt{3}$.

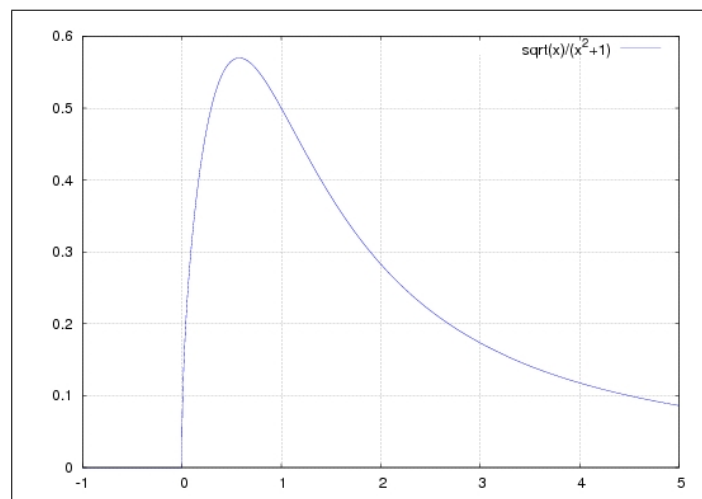


Figura 2.12: Gráfica de la función $f(x) = \frac{\sqrt{x}}{1+x^2}$

- (d) Para representar su gráfica, utilizaremos el botón [Gráficos 2D...]. Escribimos la expresión $f(x)$, el intervalo x que sea de nuestro interés (el intervalo y es opcional) y seleccionamos aquellas opciones que deseemos. El resultado será similar al de la figura 2.12.

2.3.5. Una panorámica de *wxMaxima*

Los ejemplos que se han tratado en el apartado anterior resumen buena parte de la filosofía de *wxMaxima*: por un lado, utilizar el área de entrada (o ventanas de diálogo del tipo *Resolver*) para introducir expresiones algebraicas, utilizando para ello una notación propia del lenguaje *Maxima* que es similar a la de otros lenguajes como *Matlab* o \TeX y que se detallará en próximas secciones. Y por otro lado, emplear los menús y botones disponibles para realizar operaciones sobre estas expresiones.

Como se ha comentado en la sección 2.3.2, éstos se agrupan en tres zonas a las que podemos llamar *área de menús*, *de iconos* y *de botones* ó *atajos*. Los dos últimos constituyen simplemente accesos rápidos a las entradas de menú más utilizadas.

Existen diez menús que contienen alrededor de un centenar de entradas, muchas de las cuales (en concreto las situadas en los menús "Ecuaciones", "Álgebra", "Análisis", "Simplificar", "Gráficos") tienen un carácter específico y serán detalladas en otros capítulos. En las próximas líneas, nos centraremos en estudiar los menús de propósito general relacionadas con el funcionamiento de *Maxima* y *wxMaxima*: "Archivo", "Editar", "Maxima", "Numérico", "Ayuda".

Menú "Archivo"

En él se puede acceder a distintas posibilidades relacionadas con el almacenamiento de información en el disco duro. Algunas de éstas son:

"Abrir sesión" Se puede utilizar para recuperar el trabajo que se haya realizado en una sesión anterior y se haya almacenado con la opción "Guardar Sesión"

"Guardar sesión" Almacenar en el disco duro todo el trabajo que se haya realizado (incluyendo variables, funciones, etc), tal y como se puede observar en el área de salida. El fichero resultante no contiene expresiones en lenguaje *Maxima*, sino en lenguaje Lisp.

"Archivo de lotes" Leer un fichero y procesar sucesivamente las expresiones en lenguaje *Maxima* contenidas en él. Este fichero se puede haber creado manualmente (con un editor de textos apropiado) o mediante el menú "Crear archivo de Lotes"

"Crear archivo de lotes" Volcar el trabajo que se haya realizado hacia un fichero de texto en lenguaje *Maxima*, apto para ser procesado posteriormente.

"Exportar a HTML" Almacenar la sesión en un fichero apropiado para su acceso mediante navegadores web.

"Salir" Abandonar la sesión

Menú "Editar"

"Copiar" Copiar la zona del área de salida (la consola) que esté seleccionada

"Copiar texto" Igual que la opción anterior, pero incluyendo también caracteres de salto de línea.

"Selección a entrada" Copiar en la línea de entrada el texto de la consola que esté seleccionado. Esto mismo se puede conseguir copiando y pegando en la entrada (teclas CTRL+C y CTRL+V).

"Limpiar la pantalla" Borrar los contenidos de la consola (no las variables, funciones, etc).

"Ampliar" y **"Disminuir"** Modificar el tamaño de la fuente que se utiliza para dibujar el contenido de la consola.

"Ir a la entrada" (F4) Situar el cursor en la línea de entrada y seleccionar su contenido (de forma que, si empezamos a escribir, se borrará todo éste).

"Preferencias" Configurar distintas características de *wxMaxima*, entre ellas la forma de ejecutar *Maxima*, el tipo de letra utilizado y el tipo de panel de botones: "Desactivado", "Básico" o "Completo"

Menú "Maxima"

Interactuar con el programa *Maxima* que subyace ejecutándose en el fondo y que *wxMaxima* utiliza para cualquier tipo de cálculo.

"Interrumpir" Detener los cálculos que se estén realizando en este momento

"Reiniciar Maxima" Reiniciar el programa *Maxima* subyacente.

"Limpiar memoria" Eliminar todas las variables que se hayan definido por parte del usuario

"Mostrar funciones" Mostrar las funciones que hayan sido definidas por el usuario

"Mostrar definición" Mostrar la definición de una función de usuario

"Mostrar variables" Mostrar las variables que hayan sido definidas por el usuario

"Borrar función" Eliminar una función de usuario que haya sido previamente definida

"Borrar variable" Eliminar una variable de usuario

“Conmutar pantalla de tiempo” Mostrar, para cada nueva orden, el tiempo que ésta haya tardado en ejecutarse

“Cambiar pantalla 2D” Cambia el algoritmo utilizado para representar expresiones matemáticas en la consola

“Mostrar formato T_EX” Mostrar el código T_EX asociado a la expresión anterior

Menú “Ayuda”

“Ayuda de Maxima” Muestra los libros de ayuda de *Maxima* que estén disponibles en el sistema. Estos libros se pueden navegar de forma interactiva y en cada uno de ellos se pueden realizar búsquedas de palabras clave.

“Describir” Muestra la descripción de cualquier función del sistema *Maxima* (como “factor”, “solve”, etc)

“Ejemplo” Muestra un ejemplo de uso de cualquier función de *Maxima*

“A propósito” Muestra funciones de *Maxima* similares a una palabra

“Mostrar sugerencia” Muestra una idea relacionada con el uso de *wxMaxima*

“Información de compilación” / “de error” Muestra información técnica relacionada con el funcionamiento del programa.

Bibliografía

- [1] The gnu operating system. <http://www.gnu.org>.
- [2] J. González Barahona, J. Seoane Pascual, and G Robles. *Introducción al Software Libre*. Universitat Oberta de Catalunya, 2004.
- [3] Mario Rodríguez Riotorto. Primeros pasos en maxima. <http://page.axiom-developer.org/zope/Plone/refs/books/axiom-book2.pdf>, 2006.
- [4] Richard M. Stallman. Software libre para una sociedad libre. http://osl.uca.es/jornadas/josluca06/web/cd/Contenidos/libros_y_articulos.html#software-libre-para-una-sociedad-libre.
- [5] Maxima Team. *Manual de Maxima (versión en español)*.