Mini Tutorial de wxMaxima

Edmundo Lavia
versión 0.1
Agosto 2013

1. Introducción

El CAS (*Computer Algebra System*) Maxima es un intérprete de comandos en el mismo sentido en el que lo era el sistema operativo DOS. Una vez arrancado el entorno nos provée de un prompt (ver Figura 1) en el cual podemos tipear comandos que finalizaremos con ";" y ejecutaremos presionando ENTER (o SHIFT-ENTER dependiendo de la configuración). El resultado se presentará en la línea de salida. Luego el intérprete vuelve a estar listo para la próxima entrada.

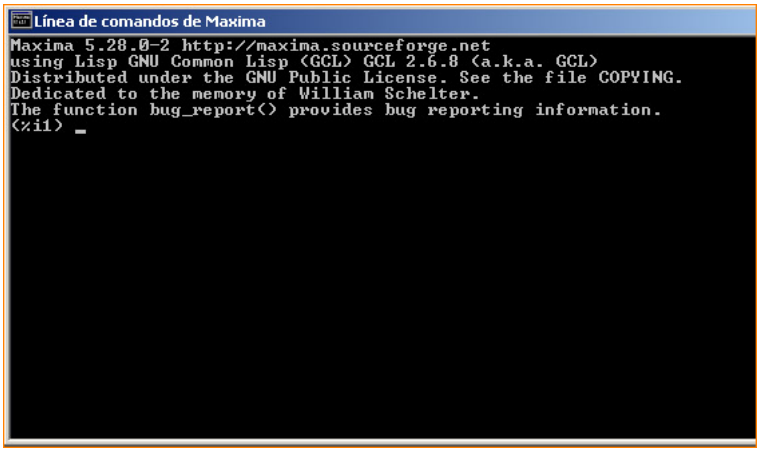

Figura 1: Línea de comandos de Maxima, bajo Windows XP.

Las capacidades de graficación son provistas por GNUPlot, que es un software también de línea de comandos pero que tiene acceso a librerías gráficas y es capaz de generar ventanas con figuras de variada índole.

Para mayor comodidad y legibilidad existen interfaces gráficas que embeben la línea de comandos en una *worksheet* (una hoja de trabajo) que puede luego guardarse como archivo de texto. Esta hoja de trabajo no es otra cosa que un listado con la secuencia original de comandos tipeados que permite, además, introducir celdas con texto, organizar las entradas con títulos y subtítulos y otras bondades. En una worksheet sólo es almacenado el historial de comandos y el texto tipeado, no los datos generados o procesados.

Una de las interfases gráficas de Maxima más usable que existe actualmente es wxMaxima. Además de la línea de mandatos provee acceso mediante menúes y barras a los comandos principales y dispone de un sistema Help de ayuda en línea muy completo. En la Figura 1 vemos la comparación entre la presentación visual del intérprete y la de la interfaz.

En lo que sigue nos referiremos al uso de wxMaxima.

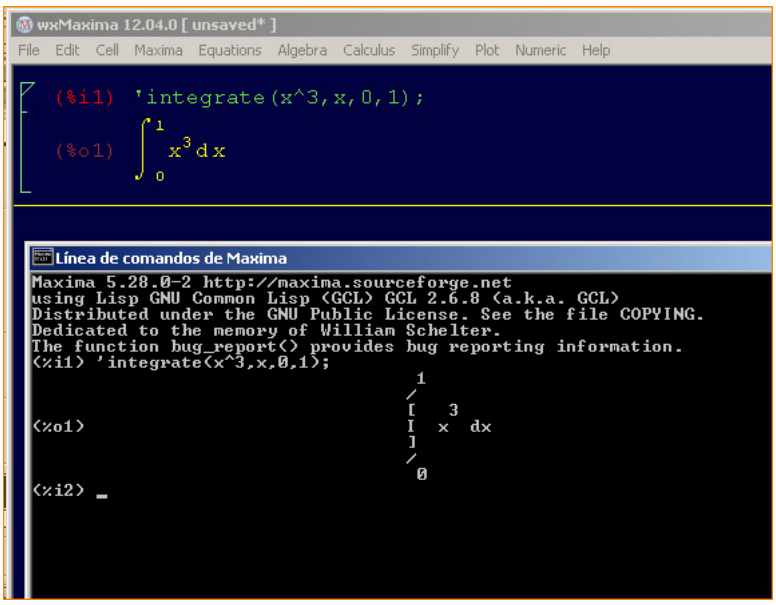

Figura 2: Ventana de wxMaxima y del intérprete bajo Windows XP mostrando la diferente capacidad de visualización. Claramente wxMaxima es más agradable al ojo.

1.1. Uso de wxMaxima. Aritmética y asignación.

Al arranque, wxMaxima inicia con una hoja de trabajo vacía. Después de un instante ya estamos en condiciones de tipear comandos. Veamos primeramente que Maxima es, ante todo, una calculadora

```
(%i4)  23*45^2+48/9-15;
```

$$(\%o4)\quad \frac{139696}{3}$$

Observamos en esta primera entrada los operadores usuales $+, -, *, /$ y $\hat{}$. Observemos ahora cómo los paréntesis alteran la precedencia de las operaciones,

```
(%i5)  (23*45^2+48)/9 -15;
```

$$(\%o5)\quad \frac{15496}{3}$$

Podemos separar comandos sucesivos con ';' de forma que ejecutando la línea se ejecutan sucesivamente todos y obtenemos tantas líneas de salida como comandos ejecutemos.

```
(%i6)  34*8 ; sqrt(15689) ; (1/5)-69/8 ;
```

$$(\%o6)\quad 272$$

$$(\%o7)\quad \sqrt{15689}$$

$$(\%o8)\quad -\frac{337}{40}$$

Maxima trabaja con precisión arbitraria y trata de conservar los números en forma exacta tanto como sea posible. Pero a veces es práctico forzar una aproximación como número decimal,

```
(%i9)  23*45^2+48/9-15 , numer;
```

$$(\%o9)\quad 46565{,}33333333334$$

Hay ciertas constantes que tienen una escritura especial. π, e, i (unidad imaginaria). Note las siguientes salidas.

```
(%i10) %pi; %pi, numer ; %i ; %e ; exp(1);
```

(%o10) π

(%o11) 3,141592653589793

(%o12) i

(%o13) e

(%o14) e

Hay que tener cuidado con la capitalizacion. No es lo mismo 'Pi' que 'pi', puesto que Maxima es sensible a las mayúsculas,

```
(%i15) %pi, numer; %Pi, numer;
```

(%o15) 3,141592653589793

(%o16) Π

En este último caso, %Pi no tiene asignado valor alguno. Claramente no es π (relación entre circunferencia y diámetro). Definimos variables que contendrán valores con el operador de asignación ':'. Sea que nos fabricamos una variable `PI_aprox`, a la cual asignamos un valor aproximado de %pi,

```
(%i17) PI_aprox:%pi, numer;
```

(%o17) 3,141592653589793

Examine lo siguiente

```
(%i18) sin(%pi); sin(PI_aprox);
```

(%o18) 0

(%o19) $1{,}2246063538223773\,10^{-16}$

¿Es correcto? La respuesta es que si, por supuesto. Simplemente seno de un número muy cercano a π no es cero, aunque está muy cerca.

1.2. Definición de funciones. Cálculo.

El operador ':', cuya acción acabamos de ver, asigna una expresión a un nombre mientras que ':=' define un operador. Sea una función de x por ejemplo,

```
(%i20) G: 4*x^3-2*x+sin(2*x);
```

(%o20) $\sin(2\,x) + 4\,x^3 - 2\,x$

El operador puede tener dependencia explícita (es lo más parecido a la definición de una función matemática) como el siguiente ejemplo,

```
(%i21) G(x,y,sigma):=exp(sigma*x)*2+atanh(2*y);
```

(%o21) $\mathrm{G}(x, y, \sigma) := \exp(\sigma\,x)\ 2 + \mathrm{atanh}(2\,y)$

Para Maxima `G` no es el mismo ente que `G(x,y,`σ`)`.

Una expresión definida como operador se puede evaluar fácilmente y someter a operaciones, como por ejemplo la composición, de manera sencilla. La `G` definida arriba siempre será una expresión de x pero `G(x,y,`σ`)` puede componerse con cualquier expresión que se utilice como entrada. Para evaluar un operador en un punto particular simplemente reemplazamos el valor en el argumento.

```
(%i23) G(2,1,3); G(x^2,sin(3*%pi*t),sigma);
```

$$(\%o23)\ \operatorname{atanh}(2) + 2\,e^{6}$$

$$(\%o24)\ 2\,e^{\sigma\,x^{2}} + \operatorname{atanh}(2\sin(3\,\pi\,t))$$

Podemos diferenciar con respecto a alguna variable y la cantidad de veces que queramos. Note que la segunda y la tercera entrada expresan el mismo hecho pero con diferente nomenclatura de variables,

```
(%i25) diff(G(x,y,sigma),y,2);diff(G(x,y,sigma),x);diff(G(x_1,x_2,x_3),x_1);
```

$$(\%o25)\ \frac{16\,y}{\left(1-4\,y^{2}\right)^{2}}$$

$$(\%o26)\ 2\,\sigma\,e^{\sigma\,x}$$

$$(\%o27)\ 2\,x\_3\,e^{x\_1\,x\_3}$$

También la expresión asignada a `G` puede diferenciarse, pero sólo con respecto a x.

```
(%i28) diff(G,x); diff(G,v);
```

$$(\%o28)\ 2\cos(2\,x) + 12\,x^{2} - 2$$

$$(\%o29)\ 0$$

La integración también es todo un capítulo aparte, pero para muchas funciones se reduce a aplicar el comando `integrate`.

```
(%i30) integrate(G(x,y,sigma),sigma);
```

$$(\%o30)\ \sigma\,\operatorname{atanh}(2\,y) + \frac{2\,e^{\sigma\,x}}{x}$$

2. Graficación

Podemos graficar fácilmente expresiones indicando como mínimo la expresión y el intervalo de graficación. Notemos que, para un gráfico en 2D, tiene que ser una expresión que dependa de una única variable. Posiblemente debido a un *bug* es necesario correr el siguiente comando una vez en la sesión de wxMaxima para que nos permita hacer *zoom* dentro de la ventana gráfica.

```
(%i31) set_plot_option(['plot_format, 'gnuplot])$
```

El siguiente ejemplo producirá una ventana de GNUPlot conteniendo el grafico de la función. Aquí no mostraremos dicha salida.

```
(%i32) plot2d(G,[x,0,10]);
```

$(\%o32)$

En la siguiente gráfica trate de hallar a ojo (haciendo zoom y *panning* en el gráfico) dónde está el cero de este polinomio para el intervalo [0,5]

```
(%i33) plot2d( x^3-2*x^2+x-3,[x,0,5]);
```

$(\%o33)$

¿Lo pudo ver? Yo lo estimo aproximadamente entre 2.15 y 2.19, ¿verdad? Ahora ejecute el comando `solve` sobre la ecuación que define polinomio=0.

```
(%i34) solve(x^3-2*x^2+x-3=0,x);
```

$$(\%o34)\ [x=\frac{\frac{\sqrt{3}\,i}{2}-\frac{1}{2}}{9\left(\frac{\sqrt{77}}{6}+\frac{79}{54}\right)^{\frac{1}{3}}}+\left(\frac{\sqrt{77}}{6}+\frac{79}{54}\right)^{\frac{1}{3}}\left(-\frac{\sqrt{3}\,i}{2}-\frac{1}{2}\right)+\frac{2}{3},x=\left(\frac{\sqrt{77}}{6}+\frac{79}{54}\right)^{\frac{1}{3}}\left(\frac{\sqrt{3}\,i}{2}-\frac{1}{2}\right)+\frac{-\frac{\sqrt{3}\,i}{2}-\frac{1}{2}}{9\left(\frac{\sqrt{77}}{6}+\frac{79}{54}\right)^{\frac{1}{3}}}+\frac{2}{3},x=\left(\frac{\sqrt{77}}{6}+\frac{79}{54}\right)^{\frac{1}{3}}+\frac{1}{9\left(\frac{\sqrt{77}}{6}+\frac{79}{54}\right)^{\frac{1}{3}}}+\frac{2}{3}]$$

Vemos que que hemos obtenido tres raíces, que es lo que corresponde a un polinomio de grado tres. También podemos ver que la primera y la segunda tienen unidades imaginarias: ¡son complejas! La raiz que descubrimos antes de manera visual tiene que ser la tercera. La salida del comando `solve`, es en realidad una lista lo cual identificamos porque se halla enmarcada entre corchetes [ ]. Es una lista de tres elementos.

Ahora, si asignamos la salida de este comando, la lista, a una variable,

```
(%i35) Lista_soluciones:solve(x^3-2*x^2+x-3=0,x);
```

$$(\%o35)\ [x=\frac{\frac{\sqrt{3}\,i}{2}-\frac{1}{2}}{9\left(\frac{\sqrt{77}}{6}+\frac{79}{54}\right)^{\frac{1}{3}}}+\left(\frac{\sqrt{77}}{6}+\frac{79}{54}\right)^{\frac{1}{3}}\left(-\frac{\sqrt{3}\,i}{2}-\frac{1}{2}\right)+\frac{2}{3},x=\left(\frac{\sqrt{77}}{6}+\frac{79}{54}\right)^{\frac{1}{3}}\left(\frac{\sqrt{3}\,i}{2}-\frac{1}{2}\right)+\frac{-\frac{\sqrt{3}\,i}{2}-\frac{1}{2}}{9\left(\frac{\sqrt{77}}{6}+\frac{79}{54}\right)^{\frac{1}{3}}}+\frac{2}{3},x=\left(\frac{\sqrt{77}}{6}+\frac{79}{54}\right)^{\frac{1}{3}}+\frac{1}{9\left(\frac{\sqrt{77}}{6}+\frac{79}{54}\right)^{\frac{1}{3}}}+\frac{2}{3}]$$

podemos acceder a cada una de las soluciones. Nos interesa en particular la tercera:

```
(%i36) Lista_soluciones[3];
```

$$(\%o36)\ x=\left(\frac{\sqrt{77}}{6}+\frac{79}{54}\right)^{\frac{1}{3}}+\frac{1}{9\left(\frac{\sqrt{77}}{6}+\frac{79}{54}\right)^{\frac{1}{3}}}+\frac{2}{3}$$

Pero para comparar con nuestro valor visual podemos pedir un valor aproximado,

```
(%i37) Lista_soluciones[3], numer;
```

$$(\%o37)\ x=2{,}17455941029298$$

¿Le parece razonable este valor?. Si no, pruebe el siguiente comando cuya funciòn es hallar raíces de una expresión en cierto intervalo.

```
(%i38) find_root(x^3-2*x^2+x-3,x,-5,5);
```

(%o38) 2,17455941029298

Bastante bien, ¿no? Podemos graficar dos o más funciones a la vez introduciéndolas en una lista, lo cual se hace encerrando el conjunto entre corchetes [ ] y separando cada elemento con una coma. Asi por ejemplo,

```
(%i39) plot2d([G,sin(3*x),sqrt(G)],[x,0,2]);
```

(%o39)

Podemos poner leyendas en los gráficos, etiquetar los ejes y salvar a disco como archivo gráfico la salida obtenido. En el ejemplo mostrado a continuación guardamos como archivo PNG (formato raster

-bitmap-). Otra opción es salvar a PostScript (formato vectorial), pero bastará para nuestros propósitos trabajar con PNG. Por supuesto en Linux las rutas de archivo son con la otra barra "/". Para dividir un comando largo en varias líneas presionamos SHIFT-ENTER donde sea necesario descender a la línea siguiente. Se evaluará con ENTER tras el ';'.

```
(%i40) plot2d([G,sin(3*x),sqrt(G)],[x,0,2],
       [legend,"G","Seno","Raiz"],[ylabel,Funciones],[xlabel,x],
       [gnuplot_term, png], [gnuplot_out_file, "c:\graph3.png"]);
```

(%o40) c : *graph3.png*

En Windows, desde la ventana de GnuPlot podemos hacer ajustes al gráfico, como controlar el aspecto de las líneas, ocultar alguna función, exportar como archivo EMF, copiar al portapapeles, etc. Gnuplot también provée acceso a otras funcionalidades a través del mouse y del teclado. Por defecto SHIFT+*Rueda de mouse* hacen panning en el gráfico y CTRL+*Rueda* hacen zoom. Con botón izquierdo puede seleccionarse un área rectangular y hacer zoom de esa área en particular. La tecla `a` provée un shortcut inmediato para volver a la ventana original del gráfico y la tecla `g` permite activar y desactivar el grid.

Para finalizar puede verse que para graficar una lista de puntos, valores [a,b], puede emplearse la opción `discrete` del siguiente modo,

```
(%i41) plot2d([discrete,[[1,2],[2,3.4],[3,4.2],[4,5]]], [style,points]);
```

(%o41)

Digamos que estos son los rudimentos mínimos necesarios para empezar a trabajar. La documentación se instala con el programa así como links a tutoriales en internet.