Manual de uso de Maxima y wxMaxima en asignaturas de cálculo diferencial

Profesor: José Antonio Vallejo Rodríguez
Facultad de Ciencias
Universidad Autónoma de San Luis Potosí (México)
e-mail:jvallejo@fciencias.uaslp.mx
URL: http://galia.fc.uaslp.mx/∼jvallejo

Semestre III Curso 2008 – 09

Introducción

Este manual sirve como complemento de las clases magistrales impartidas en la asignatura Cálculo III, en la facultad de ciencias de la Universidad Autónoma de San Luis Potosí (México).
El objetivo es que el estudiante domine las herramientas informáticas que tiene a su disposición para realizar cálculos mecánicos dentro de la disciplina del cálculo diferencial, de manera que pueda desempeñar su futuro trabajo, cualquiera que este sea, con mayor eficiencia. De hecho, el destino de nuestros egresados es, casi universalmente, el de la docencia en los distintos niveles del sistema educativo y muy pocos acaban con algún empleo directamente ligado al sector productivo. Para los primeros, el dominio de las herramientas como la que se propone en este manual (Maxima-wxMaxima) se convierte en un valor añadido a su práctica docente, además de un valioso instrumento a la hora de diseñar cursos, tareas, etc. Para los segundos, aunque son menores en número, el manejar con soltura lo que se ha dado en llamar Tecnologías de la Información es una necesidad. El mundo empresarial hace ya mucho tiempo que exige un conocimiento no sólo a nivel usuario de paquetes de ofimática, sino que demanda habilidades específicas sobre programación, análisis estadístico, control de la calidad, etc. La mayor parte del análisis de procesos en la empresa, es demasiado voluminoso y complejo como para que se pueda hacer "a mano". El uso de simuladores de flujo de datos, de paquetes de análisis de datos estadísticos, de modelos predictivos de valores bursátiles, etc., es algo cotidiano en cualquier empresa incluso de mediano tamaño, y lo que se pide a los profesionales de hoy no es que al acabar sus estudios universitarios ya tengan un dominio completo de estas herramientas, sino *que estén capacitados* para comprenderlas y utilizarlas.
Tradicionalmente, la enseñanza de las matemáticas se ha dividido en las clases "teóricas" y las clases "prácticas", aún cuando en estas últimas lo que se resolvían eran problemas académicos de escaso interés práctico. Cuando se deseaba evaluar los conocimientos "prácticos" del alumno, el recurso tradicional consistía en un examen con preguntas del mismo tipo, pero de cálculo más pesado. De esta manera, la nota del examen reflejaba la capacidad del alumno de efectuar cálculos mecánicos sin equivocarse en "las cuentas". No niego que alguien que supere un examen de este tipo no tenga capacidad de cálculo, pero sí me opongo a pensar que alguien que no lo pase sea necesariamente un mal estudiante de matemáticas[1]. Hay un límite a la capacidad que puede evaluar una prueba de este tipo: si en un examen de álgebra hay cuatro preguntas de las cuales una consiste en invertir una matriz 5×5, con una hora de tiempo, sin duda un estudiante que lo resuelva correctamente es un buen calculista. Pero, ¿y si se propone una matriz 10×10?, ¿25×25?. En este caso, de nada sirve esa capacidad de cálculo. Así de simple. Y las matrices que uno se encuentra cuando trabaja como matemático nunca son matrices reales 3×3. Esas sólo se ven en los

[1] No, no era mi caso ☺. A mi se me daban muy bien los exámenes de puro cálculo.

libros. Un Senior Network Analyst, por ejemplo, al estudiar la estructura de las interconexiones en una LAN de mediano tamaño se puede encontrar una matriz de incidencia 1000×1000 fácilmente. Pero alguien trabajando en Inteligencia Artificial, digamos en la generación de redes de conocimiento semántico a gran escala puede encontrarase con un sistema representado por un grafo de $1,5 \cdot 10^6$ nodos y $3,5 \cdot 10^6$ aristas...

Aquí es donde entran en juego los paquetes informáticos de cálculo simbólico (o CAS, de Computer Algebra System). Éstos nos liberan de la pesada carga de tener que hacer cálculos mecánicos, repetitivos, que sabemos cómo llevar a cabo pero que nos consumen un tiempo valiosísimo. Una vez que el estudiante, por ejemplo, ha comprendido la definición de derivada, su significado geométrico y algebraico, y ha entendido cómo se deducen las reglas de cálculo esenciales, no tiene ningún interés plantearle una lista de 500 derivadas para calcular. Del mismo modo que cualquier estudiante de hoy en día comprende lo que es una raíz cuadrada o cúbica, pero no se pone a estudiar los algoritmos para calcularlas (ya lo hacen las calculadoras con sólo pulsar una tecla), resulta mucho más conveniente aprender a calcular esas derivadas de manera rápida y eficaz, con menor riesgo de equivocaciones.

Por supuesto que esto no quiere decir que el uso de las computadoras vaya a sustituir al estudio tradicional, ni mucho menos. Llegados a este punto, me gustaría repetir como argumento las palabras de alguien mucho más capacitado que yo; se trata de Paul Lutus, quien en su web http://arachnoid.com afirma [2]:

> Así que existen programas que pueden hacer matemáticas a un nivel simbólico, y algunos de ellos son libres y gratuitos. Este hecho conduce a la objeción, que se escucha cada vez más frecuentemente, de que la gente utilizará programas como Maxima para evitar aprender siquiera el mímimo de matemáticas, dependiendo en su lugar del software de matemáticas simbólicas para ocultar su ignorancia.
>
> Esta objeción tiene una historia interesante. Hasta donde yo sé, se opuso en primer lugar a los libros, cuando éstos comenzaron a ser objetos de uso corriente. En aquel momento, los críticos argumentaron que la gente se haría dependiente de los libros y perderían la habilidad de memorizar cosas. Y ¿adivinan qué, niños y niñas?. Los críticos estaban en lo cierto: gracias a los libros hemos perdido la habilidad de memorizar ingentes cantidades de información trivial. En su lugar, hemos liberado algunas de nuestras preciadas neuronas para poder pensar acerca de lo que sabemos: en vez de simplemente

[2] Paul Lutus trabajó como ingeniero de la NASA: es responsable de gran parte del sistema informático del Space Shuttle, aún hoy en funcionamiento, y creó un modelo matemático del sistema solar que fue utilizado por el Jet Propulsion Laboratory en la misión a Marte de la sonda espacial Viking. Más tarde trabajó en Apple, donde en 1979 creó el programa de procesamiento de texto Apple Writer, durante 4 años elegido consecutivamente como uno de los mejores programas del momento por las más prestigiosas revistas (como Softalk). Otro procesador creado por Paul Lutus fue FreeWriter, distribuido como freeware en 1984. Más recientemente, ha creado la plataforma de desarrollo web basada en Java llamada Arachnophilia. Más información puede hallarse en su página web, http://arachnoid.com

> catalogar hechos, tenemos libertad para procesar hechos y convertirlos en nuevas ideas.
>
> Antes de intentar aplicar este argumento a las matemáticas, veamos si merece la pena. ¿Hemos ganado algo por adoptar los libros como medios de almacenamiento masivo?, ¿somos mejores o, por el contrario, nos hemos empobrecido intelectualmente por no tener que memorizar todo lo que necesitamos saber?. ¿No sería maravilloso poder disponer de una base viviente para poder comparar?, ¿un viajero en el tiempo procedente de la época medieval, alguien que sólo puede memorizar, pero que no puede procesar los hechos memorizados?.
>
> Se da la circunstancia de que hay gente que encarna perfectamente la esencia del contrargumento, personas que representan ejemplos vivientes del modelo medieval de aprendizaje. Se les conoce como "sabios idiotas". Los sabios idiotas típicos recuerdan por siempre las cosas que leen o ven, pueden recitar libros enteros que sólo han leído una vez, palabra por palabra, aparentemente son registradores perfectos de experiencias directas. Pero a pesar de sus habilidades de feria, en nuestro tiempo se les considera discapacitados porque no tienen ningún excedente de neuronas para hacer algo con los hechos que tan eficientemente almacenan. En el año 1500 se les hubiera considerado excelentes estudiantes y académicos, porque el conocimiento medieval consistía en hechos. Pero hoy en día esto difícilmente puede funcionar, porque *el conocimiento moderno consiste en ideas.*
>
> Como siempre ocurre con todo lo que no sean cuestiones triviales, las dos caras de este argumento son relevantes. Existe un cierto riesgo de que utilicemos el software matemático para convertirnos en vagos intelectuales. Pero también existe la posibilidad de que, en asociación con el software matemático y el uso de computadoras, la gente produzca muchas más matemáticas de las que podría utilizando los métodos antiguos. Gracias a que disponemos de computadoras para realizar los cálculos de bajo nivel, podemos dedicar nuestro tiempo a la adquisición de conocimiento matemático de alto nivel. *Justamente como hemos hecho con los libros.*

Efectivamente, tal y como nos recuerda Paul Lutus, es frecuente en nuestras universidades encontrar profesores que se empeñan en hacer de los estudiantes clones de aquel Ireneo Funes "el memorioso" de quien nos hablara Borges. Es comprensible que alguno de estos profesores, educado en la más estricta ideología medieval, quiera propagarla y mantenerla a toda costa, pero las necesidades de nuestros estudiantes hoy son otras bien distintas. Espero que difundiendo el uso de programas como Maxima, tarea a la que este manual no es más que una pequeña contribución, se consiga mejorar este estado de cosas.

Quisiera, por último, recomendar un par de sitios en la web, en español, para saber más sobre Maxima. En primer lugar, tenemos la página de José Manuel Mira Ros, en la Universidad de Murcia (España), que tiene una excelente introducción a Maxima, llamada "el manualico de Maxima", y a xMaxima. Se

complementa muy bien con este manual y se puede encontrar en

http://www.um.es/docencia/mira/xmaxima.html.

También es muy interesante el documento "Maxima con wxMaxima: software libre en el aula de matemáticas", elaborado por José Rafael Rodríguez Galván, del Departamento de Matemáticas de la Universidad de Cádiz (España). Rafael también es el director de la Oficina de software libre de la UCA, una excelente iniciativa que *todas* las universidades deberían imitar. El tutorial "Maxima con wxMaxima" cubre los aspectos de historia, instalación y manejo de la interfaz gráfica de Maxima y wxMaxima, que nosotros no veremos. Por tanto, también es un excelente complemento.

Pero *el* sitio en español para obtener la mejor información sobre Maxima, es la página personal de uno de sus desarrolladores: Mario Rodríguez Riotorto, un profesor de El Ferrol en Galicia (España) que ha desarrollado un tutorial ("Primeros pasos en Maxima") muchísimo más extenso y general que éste. La página es

http://www.telefonica.net/web2/biomates/.

En particular, Mario ha tenido la delicadeza de traducir para todos el tutorial interactivo Casting SPELs, desarrollado inicialmente por Conrad Barski para LISP y que Volker van Nek tradujo al lenguaje de Maxima, pero en alemán. Se puede localizar en

http://www.telefonica.net/web2/biomates/maxima/casting/index.htm

Mario, además, es el creador del paquete `draw`, una potentísima interfaz Maxima-GNUplot que tendremos ocasión de utilizar más adelante.

Índice de sesiones

Sesión 1: Funciones básicas de Maxima y wxMaxima

1.1 Maxima como calculadora

Maxima funciona como una calculadora científica, desde luego:

```
(%i1) 2+2;
```

(%o1) 4

```
(%i2) 2*5;
```

(%o2) 10

```
(%i3) log(10);
```

(%o3) $log\,(10)$

```
(%i4) float(%o3);
```

(%o4) 2,302585092994046

```
(%i5) factor(292);
```

(%o5) $2^2\,73$

Maxima trata a las constantes especiales con un % delante. Estas constantes se tratan como un símbolo. Si queremos sus valores con 15 cifras decimales, utilizamos `float` (para especificar que queremos usar números con coma flotante, los podemos también escribir como 5,0 en lugar de 5):

```
(%i6) %pi;
```

(%o6) π

```
(%i7) float(%pi);
```

(%o7) 3,141592653589793

```
(%i8) %e;
```

(%o8) e

```
(%i9) float(%e);
```

(%o9) 2,718281828459045

Maxima sabe simplificar expresiones algebraicas, como por ejemplo $log(ab) = log(a) + log(b)$. El siguiente ejemplo tambien puede hacerse mediante la tecla `Simplify (r)` de la interfaz gráfica:

```
(%i10) log(10);
```

(%o10) $log\,(10)$

```
(%i11) radcan(%);
```

(%o11) $log\,(5) + log\,(2)$

1.2 Números complejos

También podemos trabajar con números complejos. La unidad imaginaria es `%i`, y la notación es la habitual, $a + bi$, pero ¡ojo! el producto en Maxima siempre se indica por *. Si no se pone, ocurre esto:

```
(%i12) z1:2+3(%i);
```

$$Incorrect syntax : Syntax error z1 : 2 + 3($$

Maxima señala dónde está el error: entre el 3 y el (falta algo: el *

```
(%i12) z1:2+3*(%i);
```

(%o12) $3\,i + 2$

Fijémonos que para definir la variable `z1` hemos utilizado `z1:`, volveremos sobre este tema más adelante.
Como ya hemos mencionado, `%i` se trata como un símbolo, en conjunto, así que realmente podemos olvidarnos del paréntesis en `(%i)`:

```
(%i13) z2:-1-5*%i;
```

$$(\%o13) \qquad -5\,i-1$$

Una vez definidos los complejos $z1$ y $z2$ podemos calcular con ellos en la forma habitual:

```
(%i14) z1+z2;
```

$$(\%o14) \qquad 1-2\,i$$

```
(%i15) z1*z2;
```

$$(\%o15) \qquad (-5\,i-1)\,(3\,i+2)$$

Para desarrollar este producto, utilizamos `ratsimp` (de RATional SIMPlification; este comando es capaz de sumar denominadores, etc, es decir, trabaja simplificando expresiones racionales y esta, desde luego, lo es -con denominadores 1!). El mismo resultado se obtiene con la tecla `Simplify` en el entorno gráfico:

```
(%i16) ratsimp(%);
```

$$(\%o16) \qquad 13-13\,i$$

Con el cociente hay que tener más cuidado:

```
(%i17) z1/z2;
```

$$(\%o17) \qquad \frac{3\,i+2}{-5\,i-1}$$

```
(%i18) ratsimp(%);
```

$$(\%o18) \qquad -\frac{3\,i+2}{5\,i+1}$$

Esto no funciona porque la simplificación racional no da para más con una expresión ya en forma racional. Lo que queremos es la forma rectangular del complejo, y para eso está `rectform`. De paso, vemos cómo usar la variable `%on` para referirnos a la salida n (por supuesto, `%im` hará referencia a la entrada m). Si no se especifica n (o m), se hará referencia a la salida del último comando:

```
(%i19) rectform(%o18);
```

$$(\%o19) \qquad \frac{7\,i}{26} - \frac{17}{26}$$

Para acabar con los complejos, veamos algo sobre forma polar, módulo y raíces complejas:

```
(%i20) polarform(%);
```

$$(\%o20) \qquad \frac{e^{i\left(\pi - atan\left(\frac{7}{17}\right)\right)}}{\sqrt{2}}$$

```
(%i21) abs(exp^(%i*(%pi-atan(7/17))/sqrt(2)));
```

$$(\%o21) \qquad 1/\sqrt{2}$$

```
(%i22) sqrt(2+2*%i);
```

$$(\%o22) \qquad \sqrt{2\,i+2}$$

```
(%i23) rectform(%);
```

$$(\%o23) \qquad \frac{\sqrt{2^{\frac{3}{2}}-2}\,i}{\sqrt{2}} + \frac{\sqrt{2^{\frac{3}{2}}+2}}{\sqrt{2}}$$

1.3 Variables

Los cálculos anteriores también se pueden hacer de manera abstracta. Aprovechamos para remarcar cómo se definen variables mediante la asignación con ":", éstas pueden utilizarse más adelante:

```
(%i24) z:a+b*%i;
```

$$(\%o24) \qquad i\,b + a$$

```
(%i25) w:c+d*%i;
```

$$(\%o25) \qquad i\,d + c$$

```
(%i26) z+w;
```

$$(\%o26) \qquad i\,d + c + i\,b + a$$

```
(%i27) z^2;
```

$$(\%o27) \qquad (i\,b + a)^2$$

Para desarollar este tipo de expresiones tenemos el comando `expand`:

```
(%i28) expand(%);
```

$$(\%o28) \qquad -b^2 + 2\,i\,a\,b + a^2$$

Cuando ya no nos interese seguir guardando una variable, como z, podemos borrar la asignación $z = a + bi$ que habíamos hecho. De esta forma, z volverá a ser un símbolo sin significado especial. Esto se hace con el comando `remvalue`:

```
(%i29) remvalue(z);
```

$$(\%o29) \qquad [z]$$

```
(%i30) z+w;
```

$$(\%o30) \qquad z + i\,d + c$$

1.4 Algunas funciones adicionales

Las funciones trigonométricas también se comportan como cabe esperar:

```
(%i31) cos(%pi);
```

$$(\%o31) \qquad -1$$

```
(%i32) sin(%pi/2);
```

$$(\%o32) \qquad 1$$

```
(%i33) atan(sqrt(2)/2);
```

(%o33) $$atan\left(\frac{1}{\sqrt{2}}\right)$$

```
(%i34) float(%);
```

(%o34) ,6154797086703875

```
(%i35) atan(1);
```

(%o35) $$\frac{\pi}{4}$$

Ya hemos visto un ejemplo de uso del comando `expand`. Ahora, lo aplicamos a la fórmula del binomio:

```
(%i36) (a+b)^7;
```

(%o36) $$(b+a)^7$$

```
(%i37) expand(%);
```

(%o37) $$b^7+7\,a\,b^6+21\,a^2\,b^5+35\,a^3\,b^4+35\,a^4\,b^3+21\,a^5\,b^2+7\,a^6\,b+a^7$$

El comando `factor` (el mismo que utilizamos para sacar los factores primos de un número) hace lo contrario:

```
(%i38) factor(x^2+3*x+2);
```

(%o38) $$(x+1)\,(x+2)$$

```
(%i39) factor ((x+2)/(x^2-8*x+16));
```

(%o39) $$\frac{x+2}{(x-4)^2}$$

El comando `ratsimp` tratará de expandir al máximo las fracciones, y las sumará, multiplicará, etc. siempre que esto pueda hacerse:

```
(%i40) ratsimp(%);
```

$$(\%o40) \qquad \frac{x+2}{x^2-8\,x+16}$$

Una nota sobre `factor`: este comando siempre tratará de hacer la factorización en $\mathbb{Z}$, los enteros. Si la ecuación no es reducible en $\mathbb{Z}$ la dejará como está:

```
(%i41) x^2-2;
```

$$(\%o41) \qquad x^2-2$$

Los comandos que llevan un `trig` delante (o un `(tr)` en el modo gráfico) se refieren a funciones trigonométricas. Por ejemplo, `Expand (tr)` o `trigexpand` desarrolla funciones trigonométricas, mientras que `Simplify (tr)` o `trigsimp` las simplifican:

```
(%i42) trigexpand(cos(2*x)+sin(3*x));
```

$$(\%o42) \qquad -sin\,(x)^3-sin\,(x)^2+3\,cos\,(x)^2\,sin\,(x)+cos\,(x)^2$$

```
(%i43) trigsimp(-sin(x)^3-sin(x)^2+3*cos(x)^2*sin(x)+cos(x)^2);
```

$$(\%o43) \qquad -4\,sin\,(x)^3-2\,sin\,(x)^2+3\,sin\,(x)+1$$

1.5 Ejercicios

1. Escribir correctamente las siguientes expresiones y evaluarlas:
   a) $(3+5)7$
   b) $(3+5)/7$
   c) $(3+5)^28$
   d) $3+5^28$
   e) 8^27^4
   f) $\log(1+2^2)$
   g) $\cos(\pi)e^2-1$
2. Calcular $\sqrt[3]{67}$ con 15 cifras decimales.
3. Racionalizar la expresión $3/(1-\sqrt{7})$.

4. Calcular la suma $1/(4-20x^2)+x^2$.

5. Factorizar la expresión $x^5+2x^3+x^2+2$.

6. Calcular las raíces de $x^5+15x^4+85x^3+225x^2+274x+120$.

7. Comprobar que $\cos^2 a - \sin^2 a = 2\cos^2 a - 1$ (mostrando que los dos miembros de la ecuación son iguales).

8. Si $z=2-i$ y $w=-1+i$, calcular en forma binómica y polar:

   a) zw^2

   b) $\frac{2z-w}{z+w}$

9. Maxima dispone de un comando `binomial` para definir los coeficientes binomiales. Concretamente, `binomial(x,y)` devuelve

$$\binom{x}{y} = \frac{x!}{y!(x-y)!}.$$

   Calcular

$$\binom{10}{3}, \ \binom{6}{4}.$$

Sesión 2: Ecuaciones, funciones y gráficos

2.1 Ecuaciones

El comando básico para resolver ecuaciones de todo tipo es `solve`. A continuación, damos algunos ejemplos de su uso:

```
(%i1) solve(4*x^2+x=5,x);
```

$$(\%o1) \qquad [x = -\frac{5}{4}, x = 1]$$

```
(%i2) solve(x^4-9*x^2+18,x);
```

$$(\%o2) \qquad [x = -\sqrt{3}, x = \sqrt{3}, x = -\sqrt{6}, x = \sqrt{6}]$$

En algunos casos el comando no sabe muy bien cómo queremos resolver la ecuación y nos la devuelve sin cambios:

```
(%i3) sqrt(x-4)+x=6;
```

$$(\%o3) \qquad x + \sqrt{x-4} = 6$$

```
(%i4) solve(%,x);
```

$$(\%o4) \qquad [x = 6 - \sqrt{x-4}]$$

En estos casos, suele ayudar reescribir la ecuación eliminando las posibles ambigüedades (en este caso, los dos posibles signos de la raíz):

```
(%i5) solve(x-4=(6-x)^2,x);
```

$$(\%o5) \qquad [x = 5, x = 8]$$

Las ecuaciones pueden tratarse como variables en Maxima. Esto facilita mucho la labor de saber si el resultado que obtenemos con `solve` es efectivamente una solución o no, cosa que podemos hacer con `subst` como se muestra a continuación. *¡Es imprescindible comprobar siempre si el resultado obtenido es o no una solución correcta!.*

```
(%i6) eq1:x^3-x^2-x+1=0;
```

(%o6) $$x^3 - x^2 - x + 1 = 0$$

```
(%i7) solve(eq1,x);
```

(%o7) $$[x = -1, x = 1]$$

```
(%i8) subst(x=-1,eq1);
```

(%o8) $$0 = 0$$

El comando `solve` puede resolver sistemas de ecuaciones (incluso ecuaciones no lineales). Para ello, hay que pasarle una lista con las ecuaciones a resolver. Las listas siempre se dan entre corchetes `[ecuación1, ecuación2,...]` y de nuevo hay que especificar la(s) variable(s) respecto de la(s) que se quiere resolver el sistema. Las soluciones, si existen, también se dan en forma de lista:

```
(%i9) eq2:4*x^2+x=5;
```

(%o9) $$4\,x^2 + x = 5$$

```
(%i10) solve([eq1,eq2],x);
```

(%o10) $$[[x = 1]]$$

```
(%i11) solve([x+2*y=0,3*x+5*y=6],[x,y]);
```

(%o11) $$[[x = 12, y = -6]]$$

2.2 Funciones

Las funciones en Maxima se definen exactamente igual a como lo hacemos en el pizarrón, durante una clase ordinaria, utilizando `:=` para indicar que se está dando una definición. Una vez que se ha definido una función, sus valores se calculan también de la manera usual:

```
(%i12) f(x):=cos(exp(x));
```

(%o12) $f(x) := cos(exp(x))$

```
(%i13) f(2);
```

(%o13) $cos\left(e^2\right)$

```
(%i14) g(x,y):=sin(x*y);
```

(%o14) $g(x,y) := sin(x\,y)$

```
(%i15) g(x,%pi);
```

(%o15) $sin(\pi\,x)$

```
(%i16) g(1,%pi);
```

(%o16) 0

```
(%i17) g(1/2,%pi);
```

(%o17) 1

```
(%i18) g(1/2,%pi)+16;
```

(%o18) 17

Las funciones definidas a trozos no son un problema:

```
(%i19) H(x):=if x<0 then x^4-1 else 1-x^5;
```

(%o19) $H(x) := if \quad x < 0 \quad then \quad x^4 - 1 \quad else \quad 1 - x^5$

```
(%i20) H(-2);
```

(%o20) 15

```
(%i21) H(2);
```

$$(\%o21) \qquad -31$$

Las funciones también funcionan como variables, se pueden realizar sobre ellas las funciones algebraicas habituales y aplicarlas los comandos que ya conocemos:

```
(%i22) h(x,y):=(x+3)/(x^2-4*x+3);
```

$$(\%o22) \qquad h(x,y) := \frac{x+3}{x^2-4\,x+3}$$

```
(%i23) k(x,y):=(x^2-1)/(x-2)^2;
```

$$(\%o23) \qquad k(x,y) := \frac{x^2-1}{(x-2)^2}$$

```
(%i24) q(x,y):=h(x,y)+k(x,y);
```

$$(\%o24) \qquad q(x,y) := h(x,y) + k(x,y)$$

```
(%i25) q(x,y);
```

$$(\%o25) \qquad \frac{x+3}{x^2-4\,x+3} + \frac{x^2-1}{(x-2)^2}$$

El siguiente comando es para simplificar expresiones racionales. Se puede obtener desde el entorno gráfico con `Simplify (r)`:

```
(%i26) radcan(%);
```

$$(\%o26) \qquad \frac{x^4-3\,x^3+x^2-4\,x+9}{x^4-8\,x^3+23\,x^2-28\,x+12}$$

```
(%i27) factor(%);
```

(%o27) $$\frac{x^4-3\,x^3+x^2-4\,x+9}{(x-3)\,(x-2)^2\,(x-1)}$$

Recordemos que el comando `factor` tratará de factorizar el numerador y el denominador, pero sólo en el caso de que esto sea posible sobre los enteros. En este caso, el numerador no factoriza en $\mathbb{Z}$, como se puede ver con los siguientes comandos:

```
(%i28) factor(x^4-3*x^3+x^2-4*x+9);
```

(%o28) $$x^4-3\,x^3+x^2-4\,x+9$$

```
(%i29) solve(x^4-3*x^3+x^2-4*x+9=0);
```

Omitimos la salida de este comando por ser excesivamente larga, pero se recomienda que el lector ejecute la orden en su máquina para que vea el resultado (claramente no entero).
En la siguiente práctica se verá un comando que simplifica esta tarea de ver si las raíces son o no enteras mostrando sólo las raíces reales:

```
(%i30) realroots(x^4-3*x^3+x^2-4*x+9=0);
```

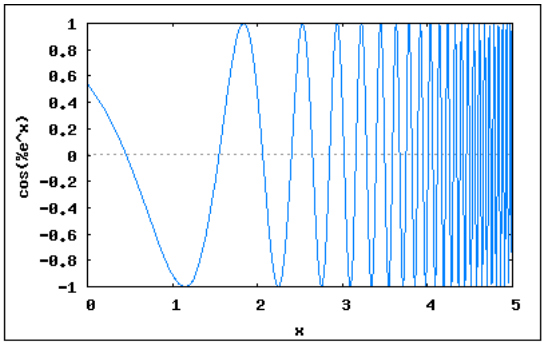
(%o30) $$[x=\frac{51146187}{33554432},x=\frac{91519551}{33554432}]$$

2.3 Gráficos de funciones

Para dibujar un gráfico de una función real de variable real, el comando básico es `plot2d`, que llamará al graficador GNUplot, un programa externo a Maxima. La sintaxis es `plot2d(función,[variable,xminimo,xmaximo])`, si x es la variable independiente. Por supuesto, también se puede hacer con el botón `Plot 2D` del entorno gráfico:

```
(%i31) plot2d(f(x),[x,0,5]);
```

(%t31)



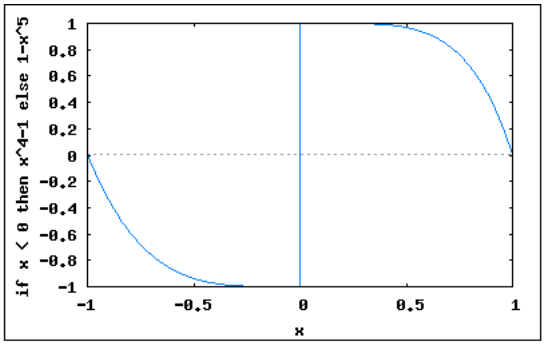
(%o31)

(%o31)

Para conseguir un gráfico en formato "inline", es decir, integrado en el documento y no en una ventana externa, se puede usar el wrapper del comando `plot2d`, en este caso el wrapper es `wxplot2d`. Un wrapper (o "envoltorio" en español) es un pequeño script que llama a un determinado comando (en este caso el comando `plot2d` de GNUplot) con ciertas opciones predeterminadas (en este caso la opción "inline") de manera que todo el proceso sea transparente para el usuario. El mismo efecto se obtiene llamando desde el entorno gráfico a `Plot 2D` y marcando la opción "inline".
Los wrappers en wxMaxima llevan todos el prefijo `wx` delante del comando, como `wxplot2d` y `plot2d`.

```
(%i32) wxplot2d(H(x),[x,-1,1]);
```

(%t32)



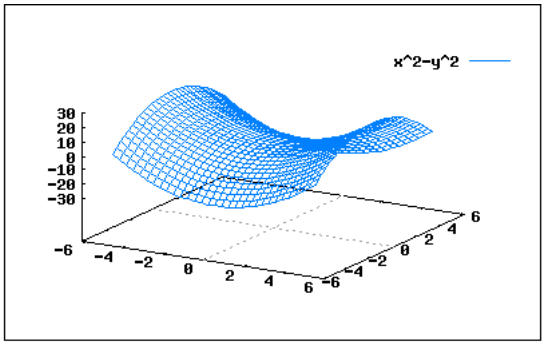
(%o32)

(%o32)

Los gráficos en 3D son completamente similares:

```
(%i33) wxplot3d(x^2-y^2,[x,-5,5],[y,-5,5]);
```

(%t33)

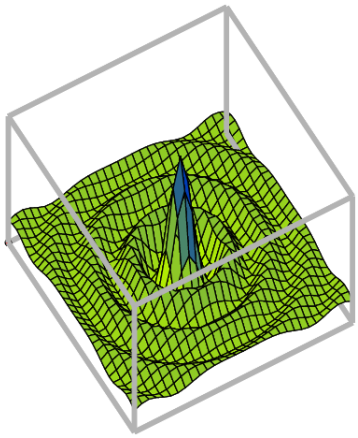


(%o33)

(%o33)

Los gráficos que se obtienen de esta forma no son interactivos, son estáticos. Existe la posibilidad de dibujarlos en el formato OpenMath, que permite operar sobre ellos con el ratón, por ejemplo, para girarlos o hacer un aumento. Para decirle a Maxima que deseamos usar el formato OpenMath usamos la variable `plot_format` del comando `plot3d`, como en el siguiente comando [3]:

```
(%i34) plot3d(sin(sqrt(x^2+y^2))/sqrt(x^2+y^2),
[x,-20,20],[y,-20,20],[plot_format,openmath]);
```

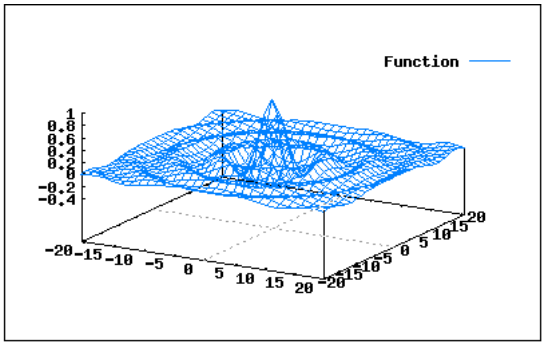


(%o34)

```
(%i35) wxplot3d(sin(sqrt(x^2+y^2))/sqrt(x^2+y^2),
[x,-20,20], [y,-20,20], [plot_format,openmath])$
```

[3] Una nota importante: los wrappers de `plot2d` y `plot3d` **no** soportan esta característica, es decir, los gráficos obtenidos con `wxplot3d` no se pueden girar dentro del documento.

(%t35)

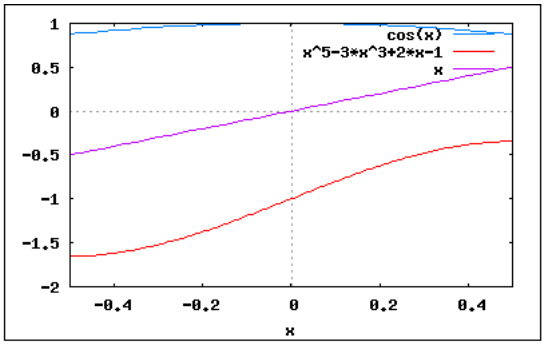


(%t35)

Se pueden dibujar varias gráficas en una misma ventana:

```
(%i36) wxplot2d([cos(x),x^5-3*x^3+2*x-1,x],[x,-0.5,0.5]);
```

(%t36)

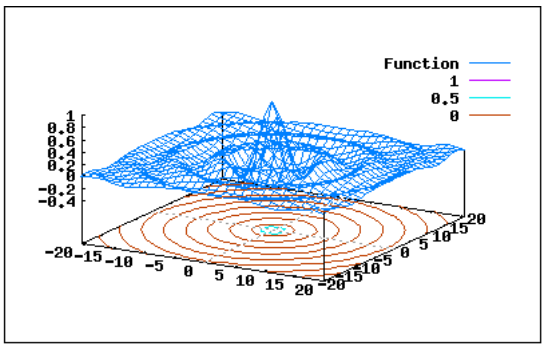


(%o36)

Si se utiliza GNUplot como formato de salida del gráfico, existe una serie de opciones muy interesantes. Se las pasamos a GNUplot mediante `[gnuplot_ preamble, "opciones"]`. Una de ellas es la posibilidad de representar las curvas de nivel de una superficie en 3D, mediante la opción `set contour`:

```
(%i37) wxplot3d(sin(sqrt(x^2+y^2))/sqrt(x^2+y^2),
[x,-20,20], [y,-20,20],
[gnuplot_preamble,  "set contour"])$
```

(%t37)

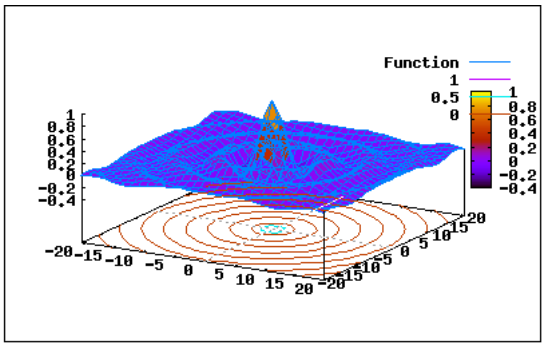


(%t37)

Otra posibilidad es la de mejorar la apariencia estética del gráfico mediante colores que varían con la altura de cada punto sobre el plano XY. Esto se consigue mediante la opción `set pm3d at s` de GNUplot. Observemos que para poner varias opciones en el preámbulo de GNUplot, estas deben ir entre comillas y separadas por punto y coma. Aquí utilizaremos `wxplot3d` para que el gráfico se integre en el documento, pero los resultados son mucho más espectaculares cuando el gráfico es interactivo, con el comando `plot3d` en lugar del `wxplot3d`.

```
(%i38) wxplot3d(sin(sqrt(x^2+y^2))/sqrt(x^2+y^2),
[x,-20,20], [y,-20,20],
[gnuplot_preamble, "set pm3d at s;set contour"])$
```

(%t38)

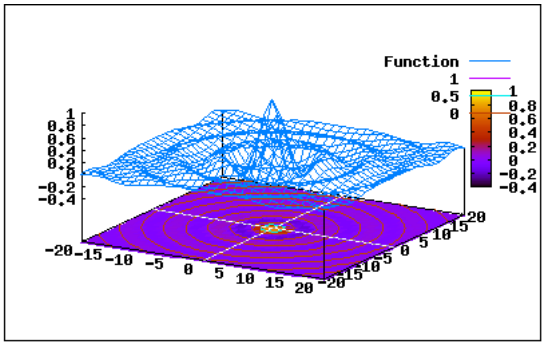


(%t38)

En la opción `set pm3d at s`, la s significa "superficie": estamos implementando las posibilidades 3D del comando al gráfico de la superficie. Otra opción es la de aplicar estas posibilidades a la proyección de la superficie en el plano XY, eso se hace con `set pm3d at b`, donde la b hace referencia a "bottom" ("fondo" en inglés):

```
(%i39) wxplot3d(sin(sqrt(x^2+y^2))/sqrt(x^2+y^2),
[x,-20,20], [y,-20,20],
[gnuplot_preamble, "set pm3d at b;set contour"])$
```

(%t39)

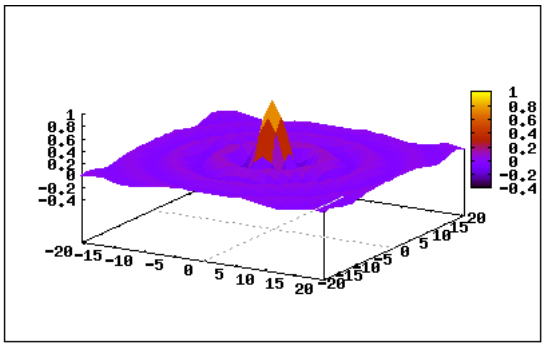


(%t39)

Otra opción más en el preámbulo de GNUplot es la de `unset surf`, que hace referencia a la posibilidad de *no* representar la malla sobre la que se construye la superficie:

```
(%i40) wxplot3d(sin(sqrt(x^2+y^2))/sqrt(x^2+y^2),
[x,-20,20], [y,-20,20],
[gnuplot_preamble, "set pm3d at s;unset surf"])$
```

(%t40)

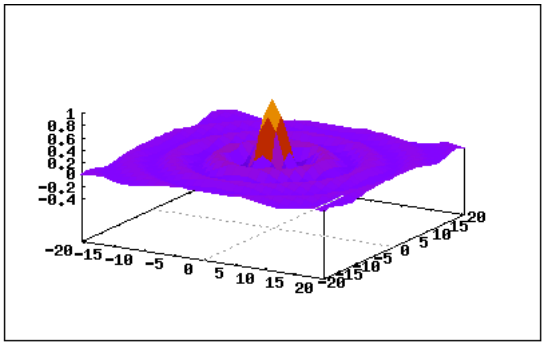


(%t40)

Por último `unset colorbox` especifica la opción de que aparezca o no la caja que indica a qué altura sobre el plano XY corresponde cada color de la gráfica:

```
(%i41) wxplot3d(sin(sqrt(x^2+y^2))/sqrt(x^2+y^2),
[x,-20,20], [y,-20,20],
[gnuplot_preamble, "set pm3d at s;unset surf;unset colorbox"])$
```

(%t41)



(%t41)

Nota importante: Existe otra posibilidad para obtener gráficos de calidad, mediante el paquete `draw`. Veremos su uso más adelante.

2.4 Ejercicios

1. Calcular las raíces de la ecuación $\sqrt{5-x} - 2x = 3$.

2. Hallar las raíces de la siguiente ecuación mediante el comando `solve`:

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

   A primera vista, el resultado parace indicar que las raíces son complejas. Esto se debe a que Maxima ha realizado el cálculo con precisión infinita. Utilizar el comando `realroots` y comprobar que, sin embargo, las tres raíces son reales.

3. Hallar todas las raíces del polinomio $x^5 - 2x^3 + x^2 - 1$.

4. Hallar las soluciones del sistema

$$\begin{aligned} -3x + 2y &= 3 \\ x - 4y &= -1 \end{aligned}$$

5. Hallar las soluciones del sistema

$$\begin{aligned} 3x^2 + 2y &= 0 \\ x + 2y^2 - y &= -1 \end{aligned}$$

6. Sea la función

$$f(x,y) = \arctan(x^2+1)\log(1+\exp(y^2x-3)).$$

Calcular $f(0,0)$, $f(-1,2)$ y $f(\pi,0)$.

7. La función paso de Heaviside está definida por

$$H(t) = \begin{cases} 0 & \text{si} \quad t<0 \\ 1 & \text{si} \quad t>0 \end{cases}.$$

a) Utilizar esta función para expresar la función $f(t)$

$$f_{t_0}(t) = \begin{cases} 0 & \text{si} \quad t<t_0 \\ 1 & \text{si} \quad t>t_0 \end{cases}.$$

Evaluar $f_{-1}(-2)$ y $f_3(2)$.

b) Utilizar la función de Heaviside para construir la función pulso cuadrado de altura k en el intervalo $[a,b]$:

$$P_k^{a,b}(t) = \begin{cases} k & \text{si} \quad a<t<b \\ 0 & \text{si} \quad t<a \ \ t>b \end{cases}$$

8. Utilizando los coeficientes binomiales, se definen los polinomios base de Bernstein de grado n. Son $n+1$ polinomios dados, para un n fijo, por

$$b_{u,n}(t) = \binom{n}{u} t^u (1-t)^{n-u}, \ u = 0,1,...,n.$$

a) Los polinomios base de grado n forman una partición de la unidad, en el sentido de que es cierta la fórmula (fácil de probar con manipulaciones teóricas)

$$b_n(t) = \sum_{u=0}^{n} b_{u,n}(t) = 1$$

Comprobar la validez de la fórmula en el caso particular de $b_4$calculando: $b_4(0)$, $b_4(1)$, $b_4(-1)$, $b_4(10)$ y $b_4(100)$. Probablemente, será útil consultar la ayuda de Maxima para recabar información acerca del comando `sum`, que calcula sumatorios.

b) Representar, en la misma ventana, los 4 polinomios base de Bernstein de grado 3 (esto es, $b_{0,3}(t)$, $b_{1,3}(t)$, $b_{2,3}(t)$, $b_{3,3}(t)$).

9. Representar en una misma ventana el paraboloide hiperbólico dado por

$$f(x,y) = x^2 - y^2$$

junto con sus curvas de nivel.

Sesión 3: Más sobre ecuaciones. Límites y derivadas

3.1 Ecuaciones II

En la sesión anterior vimos cómo resolver ecuaciones algebraicas mediante el comando `solve`. Este comando no siempre nos da la respuesta deseada, pues trata de hallar soluciones exactas mediante manipulaciones algebraicas. Cuando tal solución en forma cerrada no existe (es el caso, por ejemplo, de las ecuaciones de quinto grado, como consecuencia de un famoso teorema de Abel), `solve` no produce resultados:

```
(%i1) solve(x^5-6*x^2+8*x+3=0);
```

(%o1) $$[0 = x^5 - 6\,x^2 + 8\,x + 3]$$

Para estos casos tenemos el comando `allroots`, que encuentra las raíces de una ecuación algebraica mediante métodos de cálculo numérico:

```
(%i2) allroots(x^5-6*x^2+8*x+3=0);
```

(%o2)

$$\begin{aligned}[x = -,3049325494373391,\\ x = 1,683783821201688\,i - 1,199074650512488,\\ x = -1,683783821201688\,i - 1,199074650512488,\\ x = ,6897875494592222\,i + 1,351540925231157,\\ x = 1,351540925231157 - ,6897875494592222\,i]\end{aligned}$$

El comando `allroots` devuelve todas las raíces. Si sólo estamos interesados en las reales, podemos usar la variante `realroots`:

```
(%i3) realroots(x^5-6*x^2+8*x+3=0);
```

(%o3) $$[x = -\frac{10231839}{33554432}]$$

Para resolver sistemas de ecuaciones algebraicos (son sistemas no lineales, pero con ecuaciones únicamente racionales), Maxima dispone del comando `algsys`. Veamos un ejemplo de uso. Como siempre, las ecuaciones se pasan en una lista, igual que las incógnitas:

```
(%i4) algsys([3*x^2-y^2=6,x=y+9],[x,y]);
```

(%o4)
$$[[x=-\frac{\sqrt{255}+9}{2},y=-\frac{\sqrt{3}\,\sqrt{5}\,\sqrt{17}+27}{2}],[x=\frac{\sqrt{255}-9}{2},y=\frac{\sqrt{3}\,\sqrt{5}\,\sqrt{17}-27}{2}]]$$

La potencia de este comando es tremenda. Por ejemplo, podemos resolver sistemas algebraicos con parámetros:

```
(%i5) algsys([3*x-a*y=z,x+z=y,y^2=x],[x,y,z]);
```

(%o5) $$[[x=0,y=0,z=0],[x=\frac{a^2+2\,a+1}{16},y=\frac{a+1}{4},z=-\frac{a^2-2\,a-3}{16}]]$$

Para acabar con `algsys`, haremos notar que los coeficientes del sistema pueden ser números complejos:

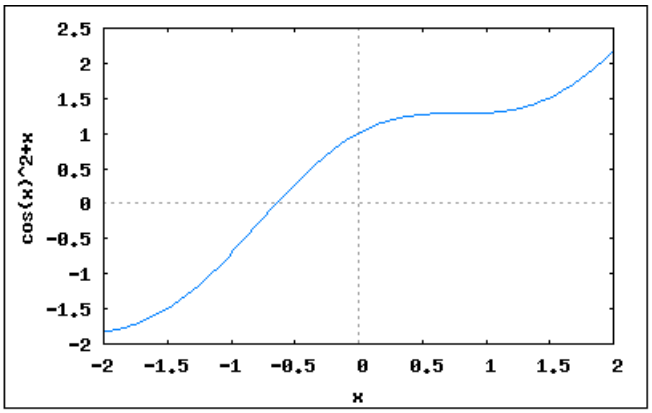
```
(%i6) algsys([(3+%i)*x-y=(1-%i)*z,x+(2-2*%i)*z=y,y^2=x],[x,y,z]);
```

(%o6) $$[[x=0,y=0,z=0],[x=\frac{9}{28\,i+45},y=\frac{3}{2\,i+7},z=\frac{33\,i+36}{27\,i+545}]]$$

Veamos ahora como hallar raíces numéricas de ecuaciones no lineales. El método que Maxima implementa es el de Newton, que necesita que le especifiquemos un intervalo en el cual se quiere localizar la raíz. Para hallar tal intervalo, lo más conveniente es representar la función gráficamente y estimar el punto de corte con el eje X:

```
(%i7) wxplot2d(cos(x)^2+x,[x,-2,2]);
```

(%t7)



(%o7)

Aquí vemos que hay una raíz entre -1 y 1. Para calcularla, haremos:

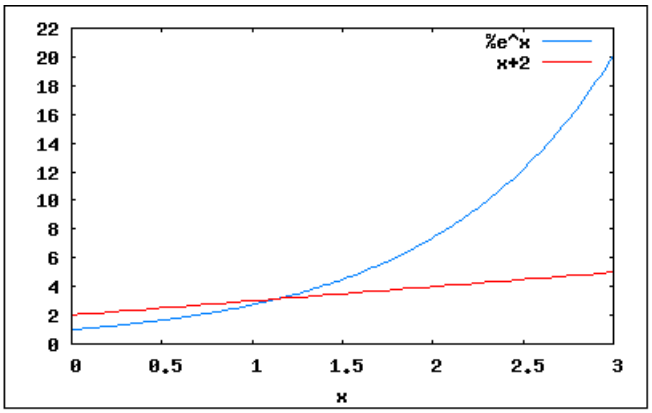
```
(%i8) find_root(cos(x)^2+x, x, -1, 1);
```

(%o8) $-,6417143708728826$

Como un ejemplo más, vamos a resolver la ecuación $e^x = x+2$, que equivale a encontrar las raíces de $e^x - x - 2 = 0$:

```
(%i9) wxplot2d([exp(x),x+2],[x,0,3]);
```

(%t9)



(%o9)

```
(%i10) find_root(exp(x)-x-2=0, x, 1, 1.5);
```

(%o10) 1,146193220620582

3.2 Límites y derivadas

El comando para hallar límites, `limit`, es uno de los más sencillos de usar. Para calcular $lim_{x \to x_0} f(x)$ se usa `limit(f(x),x,x_{0})`:

```
(%i11) f(x):=(x+a)*(x^2+b*x+c);
```

(%o11) $$f(x) := (x+a)\left(x^2 + b\,x + c\right)$$

```
(%i12) limit(f(x),x,1);
```

(%o12) $$(a+1)\,c + (a+1)\,b + a + 1$$

```
(%i13) limit(f(x),x,k);
```

(%o13) $$k^3+(b+a)\ k^2+(c+a\,b)\ k+a\,c$$

Maxima identifica límites finitos e infinitos. También puede calcular límites laterales, con los modificadores `minus` (límites por la izquierda) y `plus` (límites por la derecha):

```
(%i14) limit(1/x,x,0,minus);
```

(%o14) $$-\infty$$

```
(%i15) limit(1/x,x,0,plus);
```

(%o15) $$\infty$$

En caso de que no se usen estos modificadores y el límite sea distinto en un caso y en otro, Maxima devuelve `und` de `undefined` (indefinido):

```
(%i16) limit(1/x,x,0);
```

(%o16) $$und$$

También se pueden calcular $lim_{x\to\infty}$ y $lim_{x\to-\infty}$. Obsérvese que $-\infty$ se expresa con `minf`:

```
(%i26) limit(1/sqrt(x),x,inf);
```

(%o26) $$0$$

```
(%i27) limit((exp(x)-exp(-x))/(exp(x)+exp(-x)),x,minf);
```

(%o27) $$-1$$

El comando `limit` también opera sobre funciones de dos variables:

```
(%i28) g(x,y):=sin(x*y);
```

(%o28) $$g\left(x,y\right):=sin\left(x\,y\right)$$

```
(%i29) limit(g(x,y),x,-2);
```

(%o29) $$-sin\left(2\,y\right)$$

Vamos ahora con las derivadas. El comando para indicar derivación con respecto a una variable es `diff(funcion,variable)`, como en el ejemplo siguiente:

```
(%i31) diff(g(x,y),x);
```

(%o31) $$y\,cos\left(x\,y\right)$$

```
(%i32) diff(g(x,y),y);
```

(%o32) $$x\,cos\left(x\,y\right)$$

Se pueden calcular derivadas segundas, terceras, etc., sin más que indicar el orden de derivación a continuación de la variable:

```
(%i33) diff(g(x,y),x,2);
```

(%o33) $$-y^2\,sin\left(x\,y\right)$$

Con esto, por ejemplo, estamos en condiciones de comprobar la igualdad de las derivadas cruzadas (teorema de Schwarz):

```
(%i34) diff(diff(g(x,y),x,2),y,2);
```

(%o34) $$x^2\,y^2\,sin\left(x\,y\right)-2\,sin\left(x\,y\right)-4\,x\,y\,cos\left(x\,y\right)$$

```
(%i35) diff(diff(g(x,y),y,2),x,2);
```

(%o35) $$x^2\,y^2\,sin\left(x\,y\right)-2\,sin\left(x\,y\right)-4\,x\,y\,cos\left(x\,y\right)$$

El apóstrofo ' delante de un comando tiene el efecto de dejarlo sin evaluar. Podemos utilizar esta propiedad para escribir resultados de operaciones en forma de ecuaciones, como por ejemplo:

```
(%i36) 'diff(cos(x^2+1), x, 2) = diff(cos(x^2+1), x, 2);
```

$$(\%o36) \qquad \frac{d^2}{d\,x^2}\,cos\left(x^2+1\right) = -2\,sin\left(x^2+1\right) - 4\,x^2\,cos\left(x^2+1\right)$$

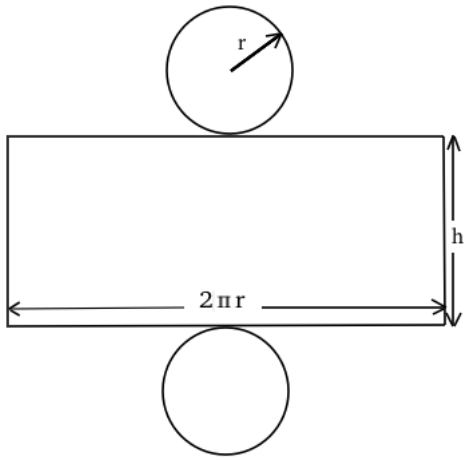
Y lo mismo ocurre con las derivadas parciales:

```
(%i37) 'diff(cos(x^2+y^2),x,1,y,2) = diff(cos(x^2+y^2),x,1,y,2);
```

$$(\%o37) \qquad \frac{\partial^3}{\partial\,x\,\partial\,y^2}\,cos\left(y^2+x^2\right) = 8\,x\,y^2\,sin\left(y^2+x^2\right) - 4\,x\,cos\left(y^2+x^2\right)$$

3.3 Aplicaciones del concepto de derivada

Vamos a calcular cuáles son las dimensiones de una lata de refresco que hacen que su área total sea mínima para un volumen de 333cl. El área de la lata es la suma de su área lateral más el área de las tapas, como se ve en la figura. Definimos la función área total $A(r,h)$ como esta suma, que será una función del radio r y de la altura h:




```
(%i38) A(r,h):=2*%pi*r^2+2*%pi*r*h;
```

$$(\%o38) \qquad A\left(r,h\right) := 2\,\pi\,r^2 + 2\,\pi\,r\,h$$

A continuación, definimos la función volumen de la lata, que también depende de r y h:

```
(%i39) V(r,h):=%pi*(r^2)*h;
```

$$(\%o39) \qquad V(r,h) := \pi\, r^2\, h$$

Este volumen no puede tomar cualquier valor, sino que debe cumplirse la restricción $V(r,h) = 333$. Esta ecuación nos permite despejar h en función de r, mediante el comando `solve`: le indicamos la ecuación que queremos resolver y respecto a qué variable (en este caso, la h):

```
(%i40) solve(V(r,h)=333,h);
```

$$(\%o40) \qquad [h = \frac{333}{\pi\, r^2}]$$

Ahora, definimos la función "área reducida", $a(r)$, obtenida sustituyendo el valor que hemos obtenido para h en función de r en $A(r,h)$. Esta nueva función sólo depende del radio:

```
(%i41) a(r):=A(r,333/(%pi*r^2));
```

$$(\%o41) \qquad a\,(r) := A\left(r, \frac{333}{\pi\, r^2}\right)$$

Con el fin de hallar sus puntos críticos, resolvemos la ecuación obtenida al igualar la derivada a cero:

```
(%i42) solve(diff(a(r),r)=0,r);
```

$$(\%o42) \qquad [r = \frac{\sqrt{3}\, 333^{\frac{1}{3}}\, i - 333^{\frac{1}{3}}}{2\, 2^{\frac{1}{3}}\, \pi^{\frac{1}{3}}}, r = -\frac{\sqrt{3}\, 333^{\frac{1}{3}}\, i + 333^{\frac{1}{3}}}{2\, 2^{\frac{1}{3}}\, \pi^{\frac{1}{3}}}, r = \frac{333^{\frac{1}{3}}}{2^{\frac{1}{3}}\, \pi^{\frac{1}{3}}}]$$

Las dos primeras soluciones son complejas y, por tanto, no nos interesan. Fíjemonos que las soluciones están dadas mediante una lista. Para quedarnos con el tercer elemento de una lista, Maxima dispone del comando `third` (obviamente, también existen los comandos `first`, `second`,...):

```
(%i43) float(third(%));
```

$$(\%o43) \qquad r = 3{,}75625258638806$$

Es decir: el tamaño óptimo de una lata de refresco (el que cuesta menos de fabricar) tiene un radio de unos 3,75 cms. La altura correspondiente resulta de sustituir este valor en la expresión de h que se dedujo de la ecuación $V(r,h) = 333$. La calculamos aplicando el comando `first` aplicado a la lista de soluciones a esta ecuación (que en realidad es una lista con un solo elemento):

```
(%i44) subst(%,first(%o40));
```

$$(\%o44) \qquad h = \frac{23{,}60123106084875}{\pi}$$

```
(%i45) float(%);
```

$$(\%o45) \qquad h = 7{,}512505172776113$$

Por tanto, las dimensiones óptimas de la lata son $r \simeq 3{,}75$ y $h \simeq 7{,}51$. Una pregunta interesante es: ¿utilizan los fabricantes de refrescos estas medidas?.

3.4 Un ejemplo de uso de la regla de la cadena

Vamos a ver un ejemplo de cómo usar las capacidades de derivación mediante la regla de la cadena que tiene Maxima. Se trata de un ejemplo clásico: mostrar que toda función de la forma $f(x+at)+g(x-at)$, con f y g funciones derivables arbitrarias, es una solución a la ecuación de ondas $a^2 u_{xx}(x,t) = u_{tt}(x,t)$.

Definimos un par de funciones auxiliares, que llamaremos h_+ y h_-. Aprovechamos la ocasión para hacer notar que los nombres de funciones y variables en Maxima pueden ser cualquier cosa, los caracteres no alfanuméricos se especifican poniendo una barra invertida \ delante:

```
(%i46) h\+(x,y):=x+a*t;
```

$$(\%o46) \qquad h_+(x,y) := x + a\,t$$

```
(%i47) h\-(x,t):=x-a*t;
```

$$(\%o47) \qquad h_-(x,t) := x - a\,t$$

Ahora, definimos nuestra hipotética solución:

```
(%i48) u(x,t):=f(h\+(x,t))+g(h\-(x,t));
```

$$(\%o48) \qquad u(x,t) := f(h_+(x,t)) + g(h_-(x,t))$$

Como f y g son funciones arbitrarias, sus derivadas son funciones arbitrarias también. Maxima sabe trabajar con ellas, pero antes debemos darles un nombre para que las pueda identificar e insertar en las espresiones correspondientes. A la derivada de f la llamaremos f' y a la de g, g'. Las derivadas segundas serán, respectivamente, f'' y g''. Para hacer esta asignación de nombres se utiliza el comando `gradef`, que permite asignar un nombre a la derivada de una función arbitraria (desconocida):

```
(%i49) gradef(f(y),f\'(y));
```

(%o49) $f\,(y)$

```
(%i50) gradef(f\'(y),f\'\'(y));
```

(%o50) $f'\,(y)$

```
(%i51) gradef(g(z),g\'(z));
```

(%o51) $g\,(z)$

```
(%i52) gradef(g\'(z),g\'\'(z));
```

(%o52) $g'\,(z)$

Comprobamos que todo funciona correctamente y que Maxima calcula las derivadas aplicando la regla de la cadena:

```
(%i53) diff(u(x,t),t,2);
```

(%o53) $a^2\,f''\,(x+a\,t)+a^2\,g''\,(x-a\,t)$

Finalmente, comprobamos que $u(x,t)$ es solución a la ecuación de ondas, viendo cuánto vale $a^2u_{xx}(x,t)-u_{tt}(x,t)$:

```
(%i54) expand(a^2*diff(u(x,t),x,2)-diff(u(x,t),t,2));
```

(%o54) 0

3.5 Ejercicios

1. Resolver el siguiente sistema mediante el comando `algsys`:

$$\begin{aligned} 3x-2y+x-t &= 2\\ y+z+2t &= 1\\ x+y-3z+t &= 0 \end{aligned}$$

¿Cuántas soluciones admite? (Es decir, ¿es compatible?, ¿determinado?).

2. Determinar las soluciones del sistema

$$\begin{aligned} 2x + y - t - 4u &= 4 \\ 3x - y + 2z - 5u &= 13 \\ x + 3y + z - t - 6u &= 7 \\ x + 2y - 3z - 2t - 2u &= -7 \end{aligned}$$

3. Resolver el siguiente sistema no lineal:

$$\begin{aligned} 2x - 3y &= z \\ ax + y^2 &= z \\ x + ay &= 3z \end{aligned}$$

   a) Tratando a como un parámetro.

   b) Tratando y como un parámetro.

4. Hallar las raíces de la ecuación no lineal $2x^3 + 6x^2 - 18x = -10$.

5. Utilizando el comando `find_root`, calcular $\sqrt[3]{70}$.

6. Resolver la ecuación $\log(2-x^2) = x^2$ con el comando `find_root`. Calcular las raíces positivas y negativas por separado.

7. Otro método muy eficiente para calcular raíces e ecuaciones no lineales es el de Newton-Raphson. Maxima lo implementa mediante el comando `newton`, cuya sintaxis es: `newton(ecuacion,variable,valor inicial, epsilon)`. Para usar este comando se requiere previamente cargar el paquete `newton1` mediante la orden `load(newton1)`. Este método requiere proporcionar una valor inicial para iniciar las iteraciones, que debe estar lo más cerca posible de la raíz que se desea calcular. El argumento `epsilon` determina a partir de qué momento cesan las iteraciones (cuando la diferencia entre dos consecutivas sea menor que él). Utilizando el comando `newton`, resolver la ecuación

$$f(x) = e^{e^x} - 5 = 0$$

   con una precisión de 0,0002.

8. Calcular los siguientes límites:

   a)
$$\lim_{x \to 0} \frac{\sin x}{x}$$

   b)
$$\lim_{x \to -1} \frac{x^2 - 1}{x^2 + 3x + 2}$$

   c)
$$\lim_{x \to 2} \frac{x^2 + 3}{x - 7}$$

d)

$$\lim_{x\to+\infty} \sqrt{x^2+2x+7} - x$$

9. Una forma de calcular límites en un punto, es la siguiente. Consideremos una función como

$$f(x,y) = \frac{x^2-y^2}{x^2+y^3},$$

definida en todo el plano menos el origen. Para calcular el límite en $(0,0)$ podemos acercarnos por rectas de la forma $y = kx$, y eso lo hacemos calculando el límite cuando $x \to 0$ de la expresión

$$f(x,kx) = \frac{x^2-k^2x^2}{x^2+k^3x^3} = \frac{1-k^2}{1+k^3x}.$$

En este caso, el cálculo proporciona

$$\lim_{x\to 0} f(x,kx) = 1-k^2$$

que depende de k, por lo que no existe el límite. Dada la función

$$g(x,y) = \frac{9x^2-y^2}{x^2+y^2},$$

determinar el valor del límite al aproximarnos:

a) Por la recta $y = x$.

b) Por la recta $y = 3x$.

c) Por la parábola $x = y^2$.

10. Calcular los límites iterados en $(0,0)$ de las funciones:

a)

$$f(x,y) = \frac{x+y}{x-y} \quad \text{¿Es continua la función en (0,0)?.}$$

b)

$$g(x,y) = x\cos\frac{1}{y}. \quad \text{¿Es continua la función en (0,0)?.}$$

11. Sea la función

$$f(x,y,z) = (1+x^2-3yz)e^{(y^2+z-sin(x))}.$$

Calcular:

a)

$$\frac{\partial^3 f}{\partial x^2 \partial z}$$

b)

$$\frac{\partial^3 f}{\partial x \partial y \partial z}$$

Evaluar estas derivadas en el punto $(-1, 0, 1)$ (mediante el comando `subst`).

12. Estudiar el problema de la lata de refresco para un volumen V arbitrario.

Sesión 4: Diferencial y derivadas parciales

4.1 Estudio de la continuidad y la derivabilidad

Definimos una función fuera de $(0,0)$, sobreentendiendo que $g(0,0)=0$:

```
(%i1) g(x,y):=x*y^2/(x^2+y^4);
```

$$(\%o1) \qquad g(x,y) := \frac{x\,y^2}{x^2+y^4}$$

Veamos la derivada parcial respecto de x en $(0,0)$ Debido a la singularidad en el origen, hay que recurrir a la definición:

```
(%i2) limit(g(t,0)/t,t,0);
```

$$(\%o2) \qquad 0$$

Y la derivada parcial respecto de y en ese punto:

```
(%i3) limit(g(0,t)/t,t,0);
```

$$(\%o3) \qquad 0$$

De hecho, podemos calcular incluso la derivada direccional en la dirección de un vector (u,v) distinto de $(0,0)$:

```
(%i4) limit(g(t*u,t*v)/t,t,0);
```

$$(\%o4) \qquad \frac{v^2}{u}$$

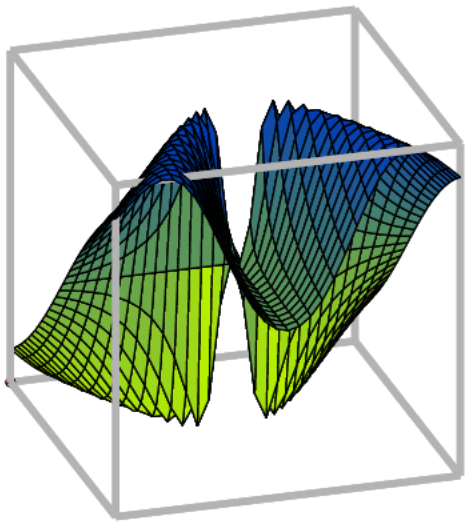
Por tanto, existen todas las derivadas direccionales en el punto $(0,0)$ y en cualquier otro. Sin embargo, la función no es continua en $(0,0)$, como se puede ver aproximándonos por parábolas $x=ay^2$:

```
(%i5) limit(g(a*y^2,y),y,0);
```

(%o5)
$$\frac{a}{a^2+1}$$

Éste, es uno de los motivos por los cuales se introduce la diferencial de una función de varias variables como generalización conveniente del concepto de derivada de funciones reales de variable real, en lugar de las derivadas parciales o direccionales.
Finalizamos el estudio obteniendo una gráfica de la función, para apreciar su comportamiento (altamente singular) en el origen.

```
(%i6) plot3d(x*y^2/(x^2+y^4),[x,-2,2],[y,-2,2],
[plot_format,openmath])
```



(%o6)

4.2 Cálculo de derivadas direccionales

Como ejemplo de cálculo, veamos cuánto vale la derivada direccional de la función $h(x,y) = x + xy^2 - y^x$ en el punto $(1,2)$ según la dirección del vector unitario $(3/34, 5/34)$. Primero lo haremos con la definición usando límites:

```
(%i7) h(x,y):=x+x*y^2-y^x;
```

(%o7) $$h\left(x,y\right):=x+x\,y^2-y^x$$

```
(%i8) limit((h(1+3*t/34,2+5*t/34)-h(1,2))/t,t,0);
```

(%o8) $$-\frac{6936\,log\left(2\right)-34680}{39304}$$

```
(%i9) ratsimp(%);
```

(%o9) $$-\frac{3\,log\left(2\right)-15}{17}$$

Ahora, haremos un cálculo mucho más general, recordando que la derivada direccional según un vector (u,v) en el punto (x,y) no es sino la diferencial de la función en el punto (x,y) actuando sobre el vector. Como la diferencial es una aplicación lineal y sus componentes como matriz son las derivadas parciales, su actuación sobre (u,v) puede verse como $dh_{(x,y)}(u,v)=uD_1h(x,y)+vD_2h(x,y)$.

```
(%i10) define(Dh1(x,y),diff(h(x,y),x));
```

(%o10) $$Dh1\left(x,y\right):=-y^x\,log\left(y\right)+y^2+1$$

```
(%i11) define(Dh2(x,y),diff(h(x,y),y));
```

(%o11) $$Dh2\left(x,y\right):=2\,x\,y-x\,y^{x-1}$$

```
(%i12) define(dh(x,y,u,v),u*Dh1(x,y)+v*Dh2(x,y));
```

(%o12) $$dh\left(x,y,u,v\right):=u\,\left(-y^x\,log\left(y\right)+y^2+1\right)+v\,\left(2\,x\,y-x\,y^{x-1}\right)$$

Comprobemos que en el punto $(1,2)$ y en la dirección $(3/34,5/34)$ obtenemos el mismo resultado que antes:

```
(%i13) dh(1,2,3/34,5/34);
```

(%o13) $$\frac{3\,\left(5-2\,log\left(2\right)\right)}{34}+\frac{15}{34}$$

```
(%i14) ratsimp(%);
```

(%o14) $$-\frac{3\,log\left(2\right)-15}{17}$$

4.3 Cálculo de la diferencial de una aplicación

Recordemos que la herramienta básica del análisis matemático es la diferencial de una función f, df. Para cada punto p del dominio de f, df_p es una aplicación lineal de $\mathbb{R}^n$ en $\mathbb{R}$, y su representación matricial es lo que llamamos vector gradiente, de manera que sus entradas son las derivadas parciales de la función: $gradf_p = (D_1 f(p), ..., D_n f(p))$. En esta práctica, veremos cómo evaluar $df_p(v)$ utilizando Maxima.

El programa dispone de un comando `grad` capaz de evaluar el gradiente de una función; para utilizarlo, hay que cargar el paquete `vect`:

```
(%i15) load("vect")$
```

Ahora bien, `grad` por defecto trabaja con coordenadas cartesianas 3D. Para especificar que queremos usar coordenadas rectangulares 2D hacemos:

```
(%i16) rectangular:[[x,y],x,y];
```

(%o16) $$[[x, y], x, y]$$

La sintaxis es clara: si quisiéramos coordenadas parabólicas en 2D sería

```
(%i16) parabolicas:[[(u^2-v^2)/2,u*v],u,v];
```

El primer corchete indica el cambio de variables desde unas coordenadas cartesianas, y a continuación se hacen explícitas las nuevas variables, en este caso (u, v). La dimensión en que se va a trabajar viene dada por el número de variables que se usen.

```
(%i17) scalefactors(rectangular);
```

(%o17) $$done$$

A partir de aquí, ya podemos definir una función y su gradiente. Para poder comparar con los métodos anteriores, utilizaremos la misma función.

```
(%i18) f:x+x*y^2-y^x;
```

(%o18) $$-y^x + x\,y^2 + x$$

El paquete `vect` es capaz de trabajar con el producto escalar directamente. En este caso, multiplicamos el gradiente por el vector (u, v).

```
(%i19) [u,v].grad(f);
```

(%o19) $[u, v].grad\left(-y^x + x\,y^2 + x\right)$

Para poder obtener las expresiones que queremos, necesitamos usar dos funciones más: `express` para expandir el gradiente en términos de las derivadas parciales, y `ev` para forzar a que éstas se evalúen para la función que hemos definido:

```
(%i20) express(%);
```

(%o20) $$v\left(\frac{d}{d\,y}\left(-y^x + x\,y^2 + x\right)\right) + u\left(\frac{d}{d\,x}\left(-y^x + x\,y^2 + x\right)\right)$$

```
(%i21) ev(%,diff);
```

(%o21) $$u\left(-y^x\,log\left(y\right) + y^2 + 1\right) + v\left(2\,x\,y - x\,y^{x-1}\right)$$

Finalmente, definimos la diferencial de h en el punto (x, y) sobre el vector (u, v):

```
(%i22) define(dh(x,y,u,v),%);
```

(%o22) $$dh\left(x, y, u, v\right) := u\left(-y^x\,log\left(y\right) + y^2 + 1\right) + v\left(2\,x\,y - x\,y^{x-1}\right)$$

4.4 Ejercicios

1. Estudiar las derivadas direccionales de las siguientes funciones, en los puntos y las direcciones indicadas:
   a) $f_1(x, y) = x^2 - 3yx + y^3$ en el punto $(0, 1)$ y la dirección $v = (1, 1)$
   b) $f_2(x, y) = x\cos(xy) + 2y^2$ en el punto $(1, 1)$ y la dirección $v = (-1, 0)$
   c) $f_3(x, y) = \log(1 + x^2 + y^2)$ en el punto $(-2, 0)$ y la dirección $v = (2, 3)$
   d) $f_4(x, y) = e^{(xy+2)}\tan(x^2 + y^2)$ en el punto $(0, 0)$ y la dirección $v = (1, 3)$.
2. Calcular las diferenciales de las funciones del ejercicio anterior en los puntos dados.
3. Comprobar que se cumple la igualdad de las derivadas cruzadas (teorema de Schwarz) para las funciones del ejercicio 1 en los puntos que se proporcionan.

4. Supongamos que $f(x, y)$ es una función diferenciable en el punto $p = (-1, 2)$ tal que su derivada direccional en p en la dirección del vector $(3/5, -4/5)$ es 8 y la derivada direccional en p según la dirección que va de p al punto $(11, 7)$ es 1. Determinar:

   a) La derivada direccional en p según la dirección $(3, -5)$.

   b) El vector gradiente en p.

5. ¿En qué dirección es igual a cero la derivada direccional de

$$f(x, y) = (x^2 - y^2)(x^2 + y^2)$$

   en el punto $(1, 1)$?. ¿Y en un punto cualquiera (x_0, y_0) del primer cuadrante?.

Sesión 5: Desarrollos de Taylor

5.1 Desarrollo de Taylor para funciones de una variable

Veamos primero cómo calcula Maxima el desarrollo de Taylor de una función de una variable alrededor de un cierto punto. El comando a emplear es `taylor`, cuya sintaxis es sumamente sencilla:

`taylor(funcion,variable,punto,grado-del-polinomio)`.

Por ejemplo,

```
(%i1) taylor(exp(x)*cos(x^2+%pi)/(x^3-1),x,0,4);
```

$$(\%o1) \qquad 1 + x + \frac{x^2}{2} + \frac{7\,x^3}{6} + \frac{13\,x^4}{24} + ...$$

Como ya sabemos, Maxima es capaz de trabajar en forma simbólica, así que podemos utilizar `taylor` para recordar el desarrollo de funciones especialmente importantes:

```
(%i2) taylor(sin(x),x,a,5);
```

```
(%o2)
```

$$sin(a) + cos(a)(x-a) - \frac{sin(a)(x-a)^2}{2} - \frac{cos(a)(x-a)^3}{6} + \frac{sin(a)(x-a)^4}{24} + \frac{cos(a)(x-a)^5}{120} + ...$$

De hecho, podemos determinar el polinomio de Taylor para cualquier función de forma abstracta:

```
(%i3) taylor(f(x),x,a,2);
```

$$(\%o3) \qquad f(a) + \left(\frac{d}{d\,x}\,f(x)\Big|_{x=a}\right)(x-a) + \frac{\left(\frac{d^2}{d\,x^2}\,f(x)\Big|_{x=a}\right)(x-a)^2}{2} + ...$$

Para funciones analíticas (aquellas que son la suma de su serie de Taylor), disponemos del comando `powerseries`, que nos da la serie de Taylor correspondiente. Para mejorar el resultado en pantalla, utilizaremos también el comando `niceindices`: su efecto es sustituir los índices por defecto que maneja Maxima (`i_1,i_2,...,i_n`) por otros más comunes como $i, j, ...$ A continuación vemos la serie de Taylor de la arcotangente:

```
(%i4) 'atan(x) = niceindices(powerseries(atan(x), x, 0));
```

$$(\%o4) \qquad atan\,(x) = \sum_{i=0}^{\infty} \frac{(-1)^{i}\, x^{2\,i+1}}{2\,i+1}$$

5.2 Animación de gráficos

Vamos a ver una posibilidad muy interesante que está disponible en wxMaxima desde la versión 0.7.4 en adelante. Se trata de hacer gráficos "animados", en el sentido de que el programa nos ofrecerá un deslizador ("slider" en inglés) que nos mostrará sucesivas gráficas dependientes de un parámetro al variar éste. El comando que crea estas "animaciones" es `wih_slider`, y aunque a primera vista su sintaxis parezca complicada, en realidad no lo es tanto. La orden básica es

```
wih_slider(parametro,lista,funcion,rangox,rangoy)
```

La explicación de cada elemento es la siguiente: `parametro` es, obviamente, el parámetro que en este ejemplo denotaremos por k, siguiendo una ancestral tradición matemática (hay otras tradiciones igualmente ancestrales que preferirán denotarlo por n...). Después viene `lista`, que como su nombre indica es una lista conteniendo los valores que tomará el parámetro. En el ejemplo que queremos analizar, vamos a representar los monomios x^k con k variando de 1 a 5, así que necesitamos crear una lista como $[1, 2, 3, 4, 5]$. Maxima tiene un comando dedicado a esta tarea: `makelist`. Su uso es:

```
makelist(formula,indice,valor-inicial,valor-final)
```

Bastante intuitivo... Se trata de definir un índice para los elementos de la lista (como por ejemplo i), definir una fórmula para determinar los valores de i (como por ejemplo $2*i+1$), y dar un valor inicial y uno final para i. Es lo mismo que las fórmulas que a todo estudiante le han aparecido alguna vez al estudiar series y analizar el comportamiento de las colas, cosas como $\sum_{i=1}^{2n} 3 * n^i$. Así, para generar una lista con los 10 primeros números impares, comenzando desde el 1, haríamos:

```
(%i5) makelist(2*j+1,j,0,9);
```

$$(\%o5) \qquad [1, 3, 5, 7, 9, 11, 13, 15, 17, 19]$$

Crear una lista con los números del 1 al 5, como nos interesa para nuestro ejemplo, es lo mas sencillo del mundo. Aquí la fórmula es simplemente "índice= i con i de 1 a 5":

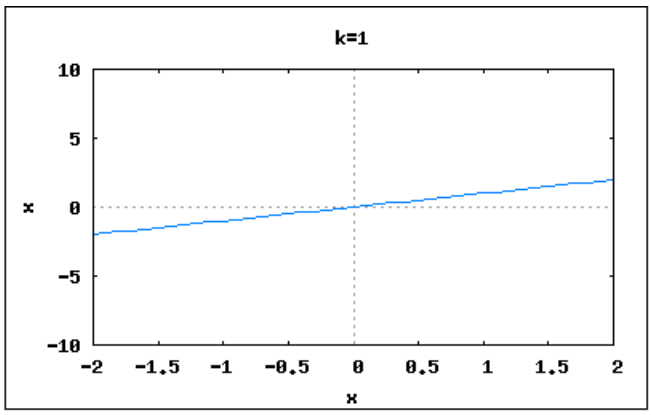
```
(%i6) makelist(j,j,1,5);
```

(%o6) $[1, 2, 3, 4, 5]$

Ya estamos en condiciones de producir nuestro primer ejemplo: representar dinámicamente los monomios x^k con k entre 1 y 5. Sólo haremos observar que el rango de la variable x se especifica como siempre, en la forma `[x,xmin,xmax]`. Especificar el rango de y no es imprescindible, el programa ajustará los valores para dar una representación adecuada, pero aquí le diremos que corte la ventana a las alturas $y = -10$ y $y = 10$:

```
(%i7) with_slider(k,makelist(j,j,1,5),x^k,[x,-2,2],[y,-10,10]);
```

(%t7)



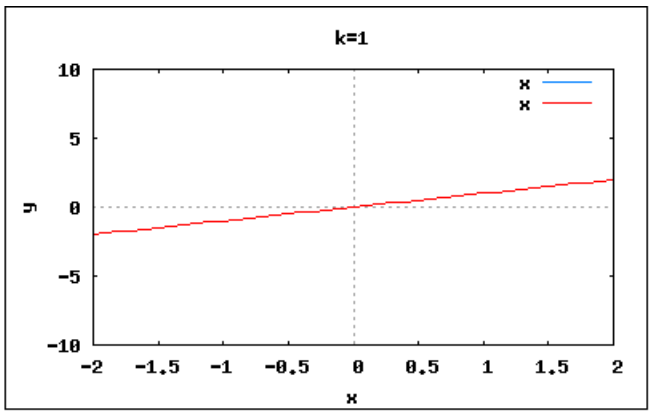
(%o7)

Para activar el gráfico, se hace click sobre él. En ese momento, veremos que en la barra de menus de wxMaxima se activa el deslizador. Haciendo click con el ratón sobre este deslizador y moviéndolo hacia la derecha se irán reproduciendo las gráficas para valores sucesivos de k, produciendo un efecto de animación.

Es posible superponer varias gráficas. Para ello, las expresamos separadas por comas dentro de unos corchetes [ ]. Podemos aprovechar esta característica para comparar las gráficas de los monomios x^k con la de una función lineal:

```
(%i8) with_slider(k,makelist(j,j,1,5),[x^k,x],[x,-2,2],[y,-10,10]);
```

(%t8)

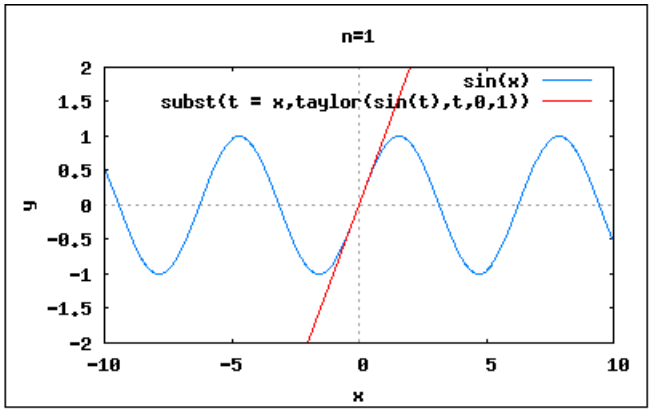


(%o8)

Veamos ahora cómo representar una curva frente a sus sucesivas aproximaciones por desarrollos de Taylor. Lo haremos en el caso particular de la función seno alrededor de $x = 0$:

```
(%i9) with_slider(n,makelist(2*j+1,j,0,5),
[sin(x),'(subst(t=x,taylor(sin(t),t,0,n)))],
[x,-10,10],[y,-2,2]);
```

(%t9)



(%o9)

Puede parecer algo extraño lo que hemos hecho aquí, pero la idea es simple. Con el comando `subst` simplemente nos quedamos con el desarrollo de Taylor de un determinado orden como una nueva expresión que dependerá de x y del valor que hayamos tomado para n. Por ejemplo, cuando $n = 3$ tenemos el desarrollo truncado de orden 3:

```
(%i10) h:subst(t=x,taylor(sin(t),t,0,3));
```

(%o10) $$x - \frac{x^3}{6}$$

Al poner el apóstrofo ' delante de `subst` estamos diciéndole a Maxima que no evalúe esta nueva expresión en ningún valor concreto de x, lo que equivale, de hecho, a tener una nueva función polinómica igual a $x - \frac{1}{6}x^3$. Al hacer esto con n variando en la lista $(1, 3, 5, 7, 9, 11)$, es como si estuviéramos pasándole a `wih_slider` la lista de polinomios $(x, x - \frac{1}{6}x^3, x - \frac{1}{6}x^3 + \frac{1}{120}x^5, ...)$ junto con $\sin(x)$ como funciones a representar.

5.3 Desarrollo de Taylor en dos variables

Maxima puede trabajar con desarrollos de Taylor con un número de variables $n > 1$. Por simplicidad, trataremos sólo el caso $n = 2$. Definamos una función de ejemplo:

```
(%i11) f(x,y):=log(1+exp(x^2*y));
```

(%o11) $$f\left(x,y\right) := log\left(1 + exp\left(x^2\,y\right)\right)$$

Ahora, para calcular el desarrollo de $f(x, y)$ alrededor del punto $(x, y) = (a, b)$, a orden 2 en x y orden 1 en y, utilizamos de nuevo `taylor`. Obsérvese como agrupamos los datos de x y de y en sus respectivos corchetes:

```
(%i12) taylor(f(x,y),[x,a,2],[y,b,1])$
```

Suprimimos la salida de este comando por ser excesivamente larga (esto se consigue con el signo del dólar `$` al final del comando anterior en lugar del habitual `;`), pero se sugiere que el lector lo ejecute en su máquina para ver el resultado.
Ahora, particularizamos al punto $(x, y) = (1, -1)$:

```
(%i13) taylor(f(x,y),[x,1,2],[y,-1,1]);
```

(%o13)

$$log\left(\left(e+1\right)\,e^{-1}\right) + \frac{y+1}{e+1} + ... + \left(-\frac{2}{e+1} + \frac{2\,\left(y+1\right)}{e^2+2\,e+1} + ...\right)\,\left(x-1\right) + \left(\frac{e-1}{e^2+2\,e+1} - \frac{\left(2\,e^2+5\,e-1\right)\,\left(y+1\right)}{e^3+3\,e^2+3\,e+1} + ...\right)\,\left(x-1\right)^2 + ...$$

Un último ejemplo. Definimos la función $g(x, y) = \exp(x^2 \sin(xy))$ y analizamos su desarrollo a orden 2 en x y en y, alrededor del punto $(x, y) = (2, 0)$:

```
(%i14) g(x,y):=exp(x^2*sin(x*y));
```

(%o14) $$g\left(x,y\right):=exp\left(x^2\,sin\left(x\,y\right)\right)$$

```
(%i15) taylor(g(x,y),[x,2,2],[y,0,2]);
```

(%o15)
$$1+8\,y+32\,y^2+...+\left(12\,y+96\,y^2+...\right)\,\left(x-2\right)+\left(6\,y+120\,y^2+...\right)\,\left(x-2\right)^2+...$$

Para dejarlo todo como un polinomio con coeficientes del tipo $x^i y^j$, utilizamos el comando `expand`:

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

(%o16) $$120\,x^2\,y^2-384\,x\,y^2+320\,y^2+6\,x^2\,y-12\,x\,y+8\,y+1$$

5.4 Desarrollos de Taylor con tres o más variables

Cuando se trabaja con funciones de más de dos variables o se van a calcular derivadas de órdenes elevados, la notación clásica usando $\frac{d}{dx}$ o $\frac{\partial}{\partial x}$ se hace inapropiada y complica mucho la apariencia visual de las fórmulas. Es mucho más conveniente utilizar la llamada notación posicional (o notación de multiíndices) para las derivadas, la misma que se usamos en las clases teóricas: una expresión como $f_{(i_1,i_2,...,i_n)}(x_1,x_2,...,x_n)$ indicará que estamos derivando i_1 veces respecto de la variable x_1, i_2 veces respecto de la variable x_2, etc. Para poder usar la notación posicional en lugar de la clásica que trae por defecto Maxima, cargaremos el paquete `pdiff`:

```
(%i17) load(pdiff)$
```

Una vez que hemos cargado el paquete `pdiff`, las derivadas de orden superior respecto de varias variables son más manejables:

```
(%i18) diff(f(x,y),x,1,y,2);
```

(%o18) $$f_{(1,2)}\left(x,y\right)$$

Lo anterior es el equivalente del clásico $\frac{\partial^3 f}{\partial x \partial^2 y}$. Otro ejemplo, esta vez calculando la derivada parcial cuarta respecto de la tercera variable (la z en este ejemplo):

```
(%i19) diff(g(x,y,z),x,0,y,0,z,4);
```

(%o19) $g_{(0,0,4)}\left(x,y,z\right)$

La expresión de los polinomios de Taylor es ahora mucho más sencilla. Incluso con funciones compuestas, pues `pdiff` aplica automáticamente la regla de la cadena, algo para lo que antes teníamos que utilizar el comando `gradef`:

```
(%i20) taylor(f(x+exp(x)),x,2,1);
```

(%o20) $f\left(e^2+2\right)+\left(f_{(1)}\left(e^2+2\right)\,e^2+f_{(1)}\left(e^2+2\right)\right)\,(x-2)+...$

Para expresar esto en la forma habitual (un polinomio en x), utilizamos `ratsimp`:

```
(%i21) ratsimp(%);
```

(%o21) $\left(e^2+1\right)\,f_{(1)}\left(e^2+2\right)\,x+\left(-2\,e^2-2\right)\,f_{(1)}\left(e^2+2\right)+f\left(e^2+2\right)$

Podemos comprobar la potencia de este paquete comprobando de nuevo que toda función de la forma $u(x,t)=f(x+at)+g(x-at)$ es una solución a la ecuación de ondas $u_{tt}(x,t)-a^2u_{xx}$, pero esta vez sin esfuerzo:

```
(%i22) u: f(x+a*t)+g(x-a*t);
```

(%o22) $f\left(x+a\,t\right)+g\left(x-a\,t\right)$

```
(%i23) ratsimp(diff(u,t,2)-a^2*diff(u,x,2));
```

(%o23) 0

Como ejemplo final, veamos cómo calcular el desarrollo de Taylor de la función $f(x,y,z)=\frac{exp(x^2+y^2+z^2)}{cos(x+y/(1+z^2))}$ alrededor del punto $(x,y,z)=(0,0,2)$ a segundo orden en x, segundo orden en y y primer orden en z (la lista `[x,0,2]` quiere decir precisamente que en la variable x desarrollamos alrededor del 0 a segundo orden, etc):

```
(%i24) taylor(exp(x^2+y^2+z^2)/cos(x+y/(1+z^2)),
[x,0,2],[y,0,2],[z,1,1]);
```

```
(%o24)
```

$$2\,e\,(z-1)+e+\ldots+\left(\frac{9\,e}{8}+2\,e\,(z-1)+\ldots\right)y^2$$
$$+\quad\left(\left(\frac{e}{2}+\frac{e\,(z-1)}{2}+\ldots\right)y+\ldots\right)x$$
$$+\quad\left(\frac{3\,e}{2}+3\,e\,(z-1)+\ldots+\left(\frac{31\,e}{16}+3\,e\,(z-1)+\ldots\right)y^2+\ldots\right)x^2+\ldots$$

Por último, expresamos el resultado como un polinomio en x, y, z:

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

(%o25)
$$\frac{\left(\left(48\,e\,x^2+32\,e\right)\,y^2+8\,e\,x\,y+48\,e\,x^2+32\,e\right)\,z+\left(-17\,e\,x^2-14\,e\right)\,y^2-24\,e\,x^2-16\,e}{16}$$

```
(%i26) expand(%);
```

(%o26)
$$3\,e\,x^2\,y^2\,z+2\,e\,y^2\,z+\frac{e\,x\,y\,z}{2}+3\,e\,x^2\,z+2\,e\,z-\frac{17\,e\,x^2\,y^2}{16}-\frac{7\,e\,y^2}{8}-\frac{3\,e\,x^2}{2}-e$$

5.5 Ejercicios

1. Calcular el polinomio de Taylor de orden 6 de la función $\sin x$ en el punto $x = 0$. Calcular el polinomio en el mismo punto de la función $\cos x$ a orden 5. Comprobar que la derivada del primero coincide con el segundo.

2. Utilizando el comando `with_slider`, representar los polinomios base de Bernstein de grado 4 en modo animación, con t variando entre 0 y 1.

3. Utilizando el comando `with_slider`, representar en una misma ventana la función $f(x) = e^{x^2+1}$ y sus aproximaciones de Taylor hasta grado 5 alrededor del punto $x = 0$.

4. Obtener el desarrollo de Taylor de la función

$$f(x, y) = \exp(x + y) + \cos(x + y)$$

alrededor del punto $(0, 0)$.

5. Demostrar que, a primer orden, la función

$$g(x, y) = \sin(x + y)$$

puede aproximarse, alrededor del punto $(0, 0)$, por la función lineal $x + y$.

6. Obtener el desarrollo de Taylor de orden 2 de la función

$$f(x,y) = \sqrt[4]{x}\sqrt[5]{y}.$$

Aproximar el valor $\sqrt[4]{1,01}\sqrt[5]{31,98}$ y dar una cota del error cometido.

7. Obtener el desarrollo de Taylor de orden 3 de la función

$$f(x,y) = x^3 + y^2 + xy^2$$

en potencias de $(x-1)$ e $(y-2)$.

Sesión 6: Cálculo de extremos

6.1 Extremos de funciones de dos variables con Hessiano no nulo

Vamos a estudiar puntos críticos de funciones de dos variables. Recordaremos el resultado básico a este respecto:
Sea $f : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$ una función contínua que admite derivadas parciales de primer y segundo orden en el punto $p \in \Omega$, el cual es un punto crítico (esto es, tal que las derivadas primeras de f se anulan en p). Sea h el Hessiano de f en p (el determinante de la matriz Hessiana en p).
Entonces:

- Si $h > 0$ y $f_{xx}(p) > 0$, el punto p es un mínimo relativo.
- Si $h > 0$ y $f_{xx}(p) < 0$, el punto p es un máximo relativo.
- Si $h < 0$, el punto p es un punto de silla.
- Si $h = 0$ no hay información (hay que hacer un análisis específico).

Comenzamos definiendo la función a estudiar:

```
(%i1) f(x,y):=log(x^2+y^2+1);
```

$$(\%o1) \qquad f(x,y) := log\left(x^2+y^2+1\right)$$

A continuación, encontramos los candidatos a puntos críticos resolviendo el sistema $\frac{\partial f}{\partial x} = 0$, $\frac{\partial f}{\partial y} = 0$:

```
(%i2) solve([diff(f(x,y),x)=0,diff(f(x,y),y)=0],[x,y]);
```

$$(\%o2) \qquad [[x = 0, y = 0]]$$

Maxima sabe cómo calcular la matriz Hessiana de una función directamente, mediante el comando `hessian`:

```
(%i3) hessian(f(x,y),[x,y]);
```

$$0errors, 0warnings$$

$$(\%o3) \qquad \begin{pmatrix} \frac{2}{y^2+x^2+1} - \frac{4\,x^2}{(y^2+x^2+1)^2} & -\frac{4\,x\,y}{(y^2+x^2+1)^2} \\ -\frac{4\,x\,y}{(y^2+x^2+1)^2} & \frac{2}{y^2+x^2+1} - \frac{4\,y^2}{(y^2+x^2+1)^2} \end{pmatrix}$$

Ahora, evaluamos la matriz Hessiana en el único candidato a punto crítico que tenemos en este caso, el $(x, y) = (0, 0)$,

```
(%i4) subst([x=0,y=0],%);
```

(%o4) $$\begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$$

```
(%i5) Hf:%;
```

(%o5) $$\begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$$

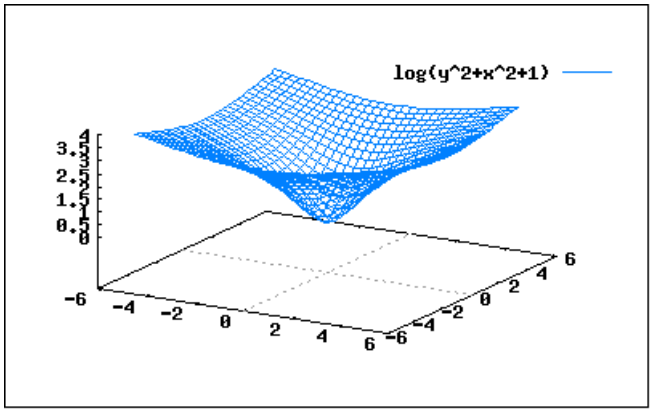
```
(%i6) determinant(Hf);
```

(%o6) $$4$$

En este caso, $h = 4 > 0$ y $f_{xx}(0,0) = 2 > 0$, luego estamos en presencia de un mínimo relativo. Lo confirmamos con la gráfica de la función:

```
(%i7) wxplot3d(log(x^2+y^2+1), [x,-5,5], [y,-5,5])$
```

(%t7)



Veamos otro ejemplo, algo más completo:

```
(%i8) g(x,y):=x^4-2*x^2*y+y-y^3;
```

(%o8) $$g\left(x,y\right) := x^4 - 2\,x^2\,y + y - y^3$$

```
(%i9) solve([diff(g(x,y),x)=0,diff(g(x,y),y)=0],[x,y]);
```

(%o9)

$$[[x=0,y=-\frac{1}{\sqrt{3}}],[x=0,y=\frac{1}{\sqrt{3}}],[x=-\frac{1}{\sqrt{3}},y=\frac{1}{3}],[x=\frac{1}{\sqrt{3}},y=\frac{1}{3}]$$
$$[x=-i,y=-1],[x=i,y=-1]]$$

Sólo nos interesarán los 4 primeros pares, que son los reales. Los llamaremos, respectivamente, $p1$, $p2$, $p3$ y $p4$. A continuación, calculamos la matriz Hessiana en esos puntos:

```
(%i10) hessian(g(x,y),[x,y]);
```

(%o10)
$$\begin{pmatrix} 12\,x^2-4\,y & -4\,x \\ -4\,x & -6\,y \end{pmatrix}$$

```
(%i11) Hfp1:subst([x=0,y=-1/sqrt(3)],%o10);
```

(%o11)
$$\begin{pmatrix} \frac{4}{\sqrt{3}} & 0 \\ 0 & \frac{6}{\sqrt{3}} \end{pmatrix}$$

```
(%i12) Hfp2:subst([x=0,y=1/sqrt(3)],%o10);
```

(%o12)
$$\begin{pmatrix} -\frac{4}{\sqrt{3}} & 0 \\ 0 & -\frac{6}{\sqrt{3}} \end{pmatrix}$$

```
(%i13) Hfp3:subst([x=-1/sqrt(3),y=1/3],%o10);
```

(%o13)
$$\begin{pmatrix} \frac{8}{3} & \frac{4}{\sqrt{3}} \\ \frac{4}{\sqrt{3}} & -2 \end{pmatrix}$$

```
(%i14) Hfp4:subst([x=1/sqrt(3),y=1/3],%o10);
```

(%o14)
$$\begin{pmatrix} \frac{8}{3} & -\frac{4}{\sqrt{3}} \\ -\frac{4}{\sqrt{3}} & -2 \end{pmatrix}$$

Y los correspondientes determinantes:

```
(%i15) h1:determinant(Hfp1);
```

(%o15) 8

```
(%i16) h2:determinant(Hfp2);
```

(%o16) 8

```
(%i17) h3:determinant(Hfp3);
```

(%o17) $-\frac{32}{3}$

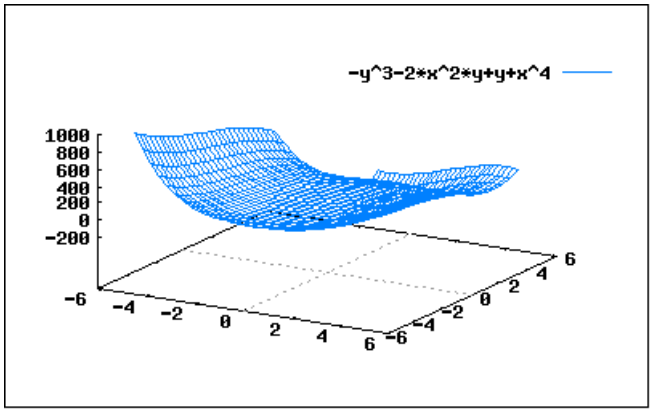
```
(%i18) h4:determinant(Hfp4);
```

(%o18) $-\frac{32}{3}$

A la vista de estos resultados, podemos afirmar que $(0, 1/\sqrt{3})$ es un máximo relativo, $(0, -1/\sqrt{3})$ es un mínimo relativo y los otros dos puntos son de silla. La gráfica de la función lo confirma:

```
(%i19) wxplot3d(x^4-2*x^2*y+y-y^3, [x,-5,5], [y,-5,5])$
```

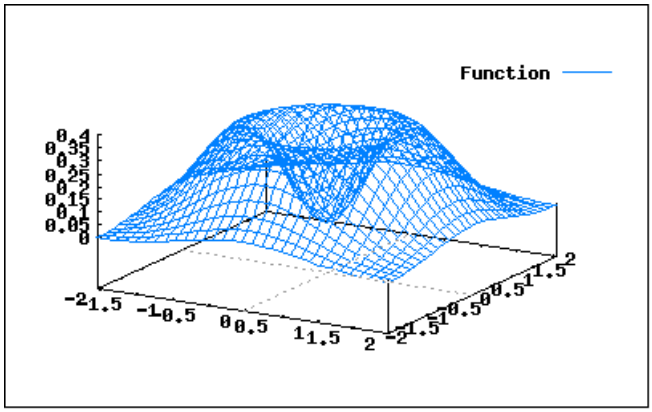
(%t19)



Como un último ejemplo, mencionaremos que hay casos en que existen infinitos puntos críticos. Por ejemplo, en el siguiente gráfico vemos el comportamiento de la función $f(x, y) = (x^2+y^2)exp(-x^2-y^2)$ (un volcán), que tiene un mínimo en $(0, 0)$ y máximos en todos los puntos de la circunferencia unidad x^2+y^2, como puede comprobarse fácilmente analizando el sistema dado por la anulación de las primeras derivadas parciales:

```
(%i20) wxplot3d((x^2+y^2)*exp(-x^2-y^2), [x,-2,2], [y,-2,2])$
```

(%t20)



6.2 Extremos de funciones de dos variables con Hessiano nulo

En esta sección, vamos a ver un par de ejemplos sobre cómo tratar el caso en que el Hessiano en los puntos candidatos a extremo se anula, de manera que el test de las segundas derivadas parciales no nos da ninguna información sobre el comportamiento de la función en ellos. Comenzaremos con una función muy sencilla:

```
(%i21) f(x,y):=x^2+y^4;
```

(%o21) $$f\left(x,y\right):=x^2+y^4$$

Calculamos los puntos candidatos a extremo:

```
(%i22) solve([diff(f(x,y),x)=0,diff(f(x,y),y)=0],[x,y]);
```

(%o22) $$[[x=0,y=0]]$$

Ahora que ya conocemos los comandos básicos para formar la matriz Hessiana en un punto, podemos hacer los cálculos de una forma mucho más eficiente, todos de una vez:

```
(%i23) Hf:subst([x=0,y=0],hessian(f(x,y),[x,y]));
```

$$0 errors, 0 warnings$$

(%o23) $$\begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix}$$

Obviamente, el determinante Hessiano en este caso es 0:

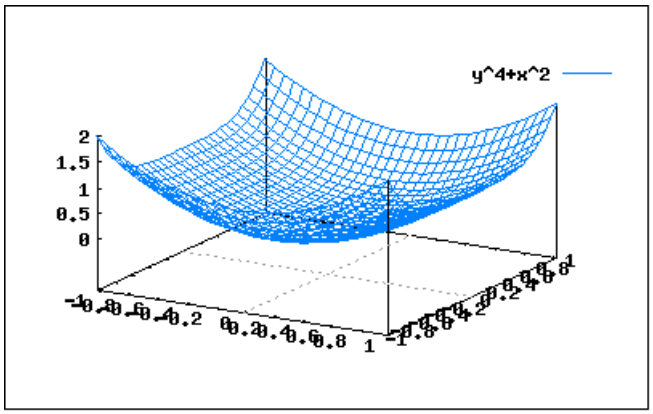
```
(%i24) determinant(Hf);
```

(%o24) $$0$$

Sin embargo, es fácil darse cuenta de que $(0,0)$ es un mínimo absoluto de $f(x,y)$. De hecho, la función es suma de dos términos positivos, x^2 e y^4, de modo que $f(x,y) \geq 0$ y el único punto en que $f(x,y) = 0$ es precisamente $(0,0)$. La gráfica de la función nos lo confirma:

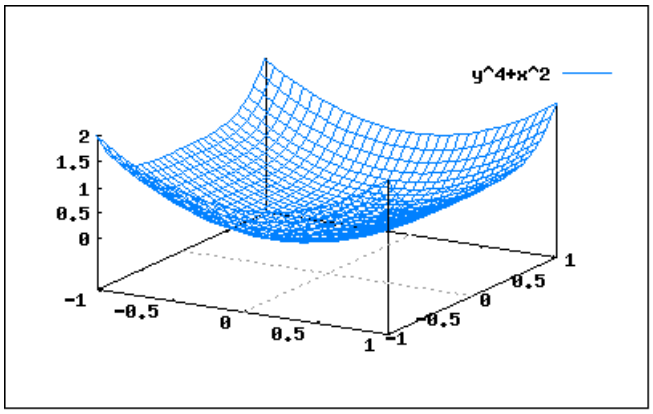
```
(%i25) wxplot3d(x^2+y^4, [x,-1,1], [y,-1,1])$
```

(%t25)



Un breve comentario sobre el gráfico. La salida por defecto de GNUplot señala unos puntos sobre los ejes X e Y, son los llamados "tics". En este caso, por defecto se indica una marca a intervalos de longitud 0.2, lo cual hace que se aproximen mucho en el gráfico dificultando la legibilidad. Para corregir esto, vamos a utilizar las opciones de GNUplot. Para pasarle opciones a GNUplot desde Maxima, recordemos (lo vimos en la página 24) que debemos incluirlas mediante una lista (como todo en Maxima) con el siguiente formato: `[gnuplot_preamble,"opcion1;...;opcionk"]` (obsérvese que las opciones se separan por punto y coma ;).
En este caso, las opciones que nos interesan son `set xtics` y `set ytics`, que controlan la apariencia de los "tics". Concretamente, vamos a indicarle a GNUplot que queremos los "tics" espaciados a intervalos de longitud 0.5, con lo que mejorará mucho la visualización de la gráfica:

```
(%i26) wxplot3d(x^2+y^4,[x,-1,1],[y,-1,1],
[gnuplot_preamble,"set xtics 0.5;set ytics 0.5"])$
```



(%t26)

Veamos un segundo ejemplo. Consideremos la función

```
(%i27) g(x,y):=x^3-y^2;
```

(%o27) $$g\left(x,y\right):=x^3-y^2$$

que tiene un único punto crítico en $(0,0)$

```
(%i28) solve([diff(g(x,y),x)=0,diff(g(x,y),y)=0],[x,y]);
```

(%o28) $$[[x=0,y=0]]$$

Este único punto es degenerado:

```
(%i29) Hg:subst([x=0,y=0],hessian(g(x,y),[x,y]));
```

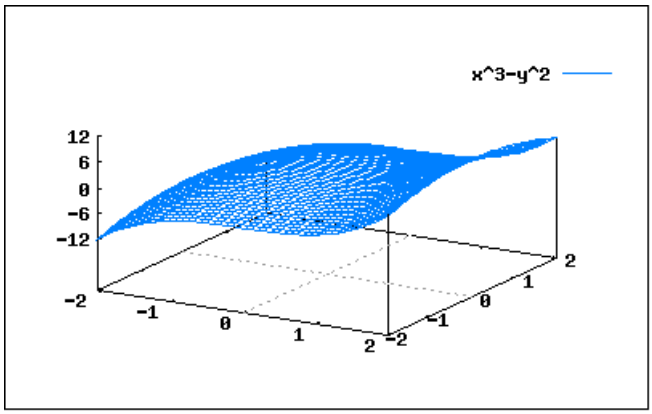
(%o29) $$\begin{pmatrix}0 & 0\\ 0 & -2\end{pmatrix}$$

```
(%i30) determinant(Hg);
```

(%o30) $$0$$

Para tener una idea de lo que está pasando, nos apoyaremos en la gráfica de la función. Esta vez, añadimos una nueva opción para GNUplot, se trata de `grid`, que determina la densidad de la malla con la que se hará la gráfica. Por defecto, la malla se establece con un tamaño de 30×30, y la aumentaremos a 45×45. También hemos modificado los "tics" del eje Z:

```
(%i31) wxplot3d(x^3-y^2,[x,-2,2],[y,-2,2],[grid,45,45],
[gnuplot_preamble,"set xtics 1;set ytics 1;set ztics 6"])$
```



(%t31)

De la gráfica se ve que $(0,0)$ tiene todo el aspecto de un punto de silla. Esta sospecha la podemos confirmar estudiando el corte de la gráfica de $g(x,y)$ con el plano $y=0$, esto es, la curva $g(x,0)=x^3$. Para ello, primero cargamos el paquete `draw`, que nos permitirá hacer cosas como representar simultáneamente dos gráficos en paramétricas (observemos que el plano $y=0$ no se puede poner en forma explícita como un grafo del tipo $z=f(x,y)$):

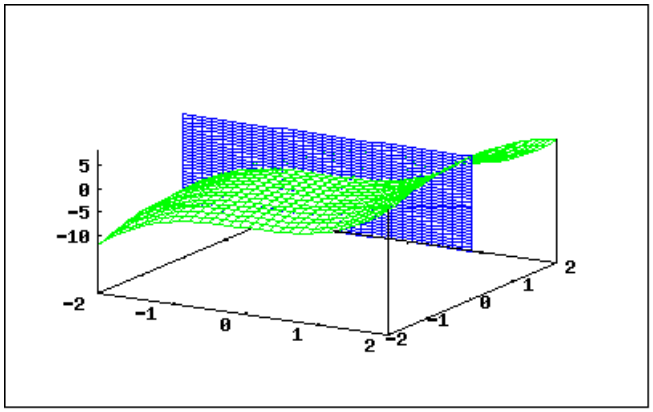
```
(%i32) load(draw)$
```

A continuación, hacemos uso de las opciones del magnífico paquete `draw`. Es imposible hacer justicia a las bondades de este paquete en unas pocas líneas, es sencillamente increíble lo que se puede hacer con él. En primer lugar, con el fin de tener gráficos "in line", utilizaremos el "wrapper" `wxdraw3d`. Podemos especificar varias superficies 3D para representarlas simultáneamente mediante un comando `parametric_surface` para cada una de ellas. El formato es

```
parametric_surface(x(u,v),y(u,v),z(u,v),u,umin,umax,v,vmin,vmax),
```

donde `umin` y `umax` son los valores mínimo y máximo que toma u, y lo mismo para v. La opción `color` determina (obviamente) el color de la superficie y con `surface_hide` haremos que las superficies sean opacas, para que se pueda apreciar el contorno de la intersección. Por último, las opciones `xtics` e `ytics` son las que ya conocemos:

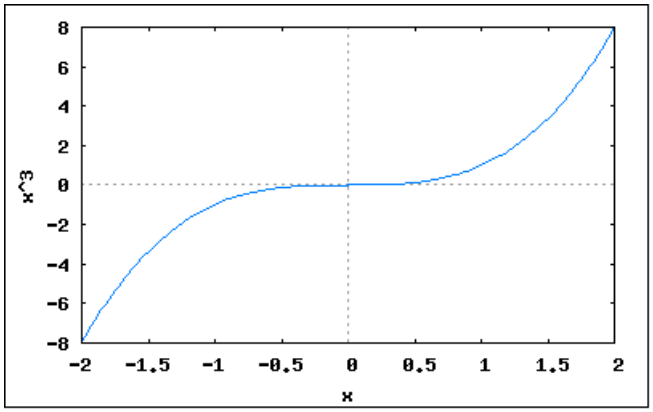
```
(%i33) wxdraw3d(color=green,
parametric_surface(u,v,u^3-v^2,u,-2,2,v,-2,2),
color=blue,
parametric_surface(u,0,v,u,-2,2,v,-12,8),
surface_hide=true,
xtics=1,ytics=1);
```



(%t33)

Como vemos en este gráfico, la intersección ciertamente indica un punto de silla. Podemos verlo definitivamente claro dibujando el contorno de esta intersección:

```
(%i34) wxplot2d(g(x,0),[x,-2,2]);
```

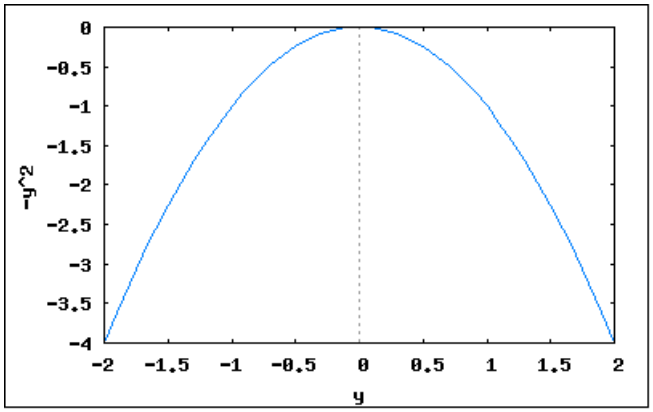


(%t34)

Cortando con el plano $x = 0$ lo que se obtiene es una parábola:

```
(%i35) wxplot2d(g(0,y),[y,-2,2]);
```

(%t35)



6.3 Extremos condicionados. Multiplicadores de Lagrange

Supongamos que queremos resolver el siguiente problema de extremos condicionados:
Optimizar la función $f(x,y) = x + y$ sujeta a las condición $xy = 4$ (es decir, sujeta a $g(x,y) = 0$, donde $g(x,y) = xy - 4$) Definimos la función de Lagrange (o Lagrangiano) asociada al problema:

```
(%i36) L(x,y,a):=x+y-a*(x*y-4);
```

(%o36) $$L\left(x,y,a\right) := x + y + \left(-a\right)\,\left(x\,y - 4\right)$$

Planteamos las condiciones *necesarias* de extremo, que son las de la anulación del gradiente del Lagrangiano:

```
(%i37) solve([diff(L(x,y,a),x)=0,
diff(L(x,y,a),y)=0,
diff(L(x,y,a),a)=0],
[x,y,a]);
```

(%o37) $$[[x = -2, y = -2, a = -\frac{1}{2}], [x = 2, y = 2, a = \frac{1}{2}]]$$

Ahora, analizaremos las condiciones *suficientes* de extremo. Para ello, recurriremos al método del Hessiano. Recordemos este método, nos dice:
Dado un punto crítico candidato a extremo (x_0, y_0, a_0), debemos calcular la forma cuadrática asociada a la matriz Hessiana *reducida* de $L(x,y,a)$ (es decir, sólo se calculan las derivadas segundas respecto de las variables (x,y)),

evaluarla en (x_0, y_0, a_0) y restringirla al subespacio dado por el núcleo de la matriz jacobiana de las ligaduras evaluada en (x_0, y_0) (como en este caso no hay más que una ligadura g, sólo hay que restringir la forma cuadrática asociada a $Hess(L(x_0, y_0, a_0))$ al subespacio dado por $(\left.\frac{\partial g}{\partial x}\right|_{(x_0,y_0)} = 0, \left.\frac{\partial g}{\partial x}\right|_{(x_0,y_0)} = 0)$).
Entonces, se tiene que:

- Si esta forma cuadrática restringida es definida negativa, el punto (x_0, y_0) es un máximo local estricto.
- Si esta forma cuadrática restringida es definida positiva, el punto (x_0, y_0) es un mínimo local estricto.
- Si esta forma cuadrática restringida es indefinida, el punto (x_0, y_0) es un punto de silla.

Calculamos, pues, la matriz Hessiana reducida:

```
(%i38) hessian(L(x,y,a),[x,y]);
```

$$0errors, 0warnings$$

(%o38)
$$\begin{pmatrix} 0 & -a \\ -a & 0 \end{pmatrix}$$

```
(%i39) HL:%;
```

(%o39)
$$\begin{pmatrix} 0 & -a \\ -a & 0 \end{pmatrix}$$

Ahora, la evaluamos para cada punto candidato, $p_1 = (-2, -2, -\frac{1}{2})$ y $p_2 = (2, 2, \frac{1}{2})$:

```
(%i40) HLp1:subst(a=-1/2,HL);
```

(%o40)
$$\begin{pmatrix} 0 & \frac{1}{2} \\ \frac{1}{2} & 0 \end{pmatrix}$$

```
(%i41) HLp2:subst(a=1/2,HL);
```

(%o41)
$$\begin{pmatrix} 0 & -\frac{1}{2} \\ -\frac{1}{2} & 0 \end{pmatrix}$$

Con esto, ya podemos calcular las formas cuadráticas asociadas a la matriz Hessiana reducida en los puntos p_1 y p_2. Maxima trata a las matrices como listas, una matriz fila $[u, v]$ se define simplemente como:

```
(%i42) U:[u,v];
```

(%o42) $[u, v]$

Y un vector columna se puede obtener trasponiendo una fila, con el comando `transpose`:

```
(%i43) Ut:transpose(U);
```

(%o43) $\begin{pmatrix} u \\ v \end{pmatrix}$

Maxima sabe automáticamente cómo multiplicar matrices. Sólo hay que decirle que emplee el producto matricial, y eso se hace escribiéndolo como un punto normal . en lugar de con el asterisco *:

```
(%i44) QLp1:U.HLp1.Ut;
```

(%o44) uv

```
(%i45) QLp2:U.HLp2.Ut;
```

(%o45) $-uv$

Estas expresiones QL_{p_1} y QL_{p_2} son indefinidas, pero eso no quiere decir que p_1 y p_2 sean puntos de silla, pues recordemos que hay que estudiar las formas cuadráticas *restringidas* al núcleo del gradiente evaluado en (x_0, y_0) de la ligadura g. Estos gradientes (uno para el punto $(-2, -2)$ y otro para el $(2, 2)$) están dados por:

```
(%i46) gradg:[diff(x*y-4,x),diff(x*y-4,y)];
```

(%o46) $[y, x]$

```
(%i47) gradgp1:subst([x=-2,y=-2],gradg);
```

(%o47) $[-2, -2]$

```
(%i48) gradgp2:subst([x=2,y=2],gradg);
```

(%o48) $[2, 2]$

y sus respectivos núcleos los calculamos haciéndolos actuar sobre un vector arbitrario (u, v) e igualando a cero:

```
(%i49) Eq1:gradgp1.Ut;
```

(%o49) $-2v - 2u$

```
(%i50) solve(Eq1,u);
```

(%o50) $[u = -v]$

Esto nos dice que la solución en el caso del primer punto $(x, y) = (-2, -2)$ está dada por $v = -u$. Para el segundo punto:

```
(%i51) Eq2:gradgp2.Ut;
```

(%o51) $2v + 2u$

```
(%i52) solve(Eq2,u);
```

(%o52) $[u = -v]$

Ahora, sustituímos la ecuación de la restricción obtenida en el primer caso, $v = -u$, en la expresión de la forma cuadrática QL_{p_1} y estudiamos cómo está definida:

```
(%i53) subst(v=-u,QLp1);
```

(%o53) $-u^2$

Claramente, esto es definido negativo y, por tanto, $(-2, -2)$ es un máximo local estricto. Para el caso de $(2, 2)$ tenemos:

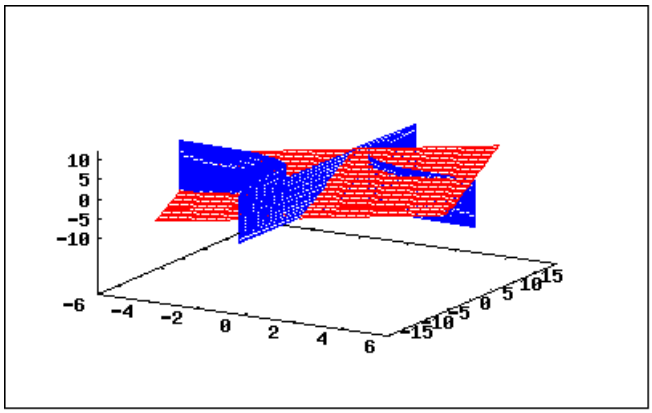
```
(%i54) subst(u=-v,QLp2);
```

(%o54) v^2

Que es definido positivo, luego $(2,2)$ es un mínimo local estricto.
Este ejemplo presenta una característica curiosa: el valor de la función en el máximo local es $f(-2,-2) = -4$, mientras que en el mínimo local es $f(2,2) = 4$, lo cual parece una contradicción. Sin embargo, no hay nada extraño aquí, pues se trata de extremos *locales*. Una gráfica nos ayudará a comprender la geometría del problema (si el paquete `draw` no estuviera cargado, recuerde cargarlo previamente a su uso):

```
(%i55) wxdraw3d(color=red,
parametric_surface(u,v,u+v,u,-6,6,v,-6,6),
color=blue,
parametric_surface(u,4/u,v,u,-6,6,v,-6,6),
surface_hide=true);
```

(%t55)



6.4 Ejercicios

1. Determinar el carácter de los puntos críticos de las siguientes funciones:

   a)
   $$f_1(x,y) = x^2 + y^4$$

   b)
   $$f_2(x,y) = x^4 + y^4 - 2(x-y)^2$$

   c)
   $$f_3(x,y) = x^3 + y^3 + 2x^2 + 4y^2$$

   d)
   $$f_4(x,y) = e^{x^2+y^2+1}$$

e)

$$f_5(x,y) = x^2y + y^2x$$

2. Comprobar gráficamente que aunque $g(x,y) = y^3 + x^3$ no presenta extremos globales, si lo hace cuando nos restringimos a la elipse $x^2 + \frac{y^2}{2} = 1$.

3. Resolver el problema de extremos condicionados dado por la función

$$f(x,y) = 60x + 90y - 2x^2 - 3y^2$$

sujeta a la ligadura

$$2x + 4y = 68.$$

4. Resolver el problema de extremos condicionados dado por la función

$$f(x,y) = x^2 + y^2$$

sujeta a la ligadura

$$x^2 + y = 4.$$

Sesión 7: Funciones inversas e implícitas

7.1 Inversas de funciones reales de una variable real

Supongamos que queremos hallar la derivada de la función inversa de la función $y = f(x)$ definida por

$$y = \frac{5}{3}x + 1.$$

Obviamente, lo primero que nos pasa por la cabeza es despejar x en función de y para luego derivar la función resultante $x = g(y)$. En este caso, despejar x es fácil:

```
(%i1) solve(5*x/3+1-y,x);
```

(%o1) $$[x = \frac{3\,y - 3}{5}]$$

De aquí, derivando:

```
(%i2) 'diff(x,y)=diff((3*y-3)/5,y);
```

(%o2) $$\frac{d}{d\,y}\,x = \frac{3}{5}$$

Este proceso es muy sencillo cuando es fácil invertir en forma explícita la función $y = f(x)$. Pero esto no siempre es posible: pensemos por ejemplo en la función

$$y = f(x) = \frac{atan(1 + e^{sin(x)})}{3 - log(1 + x^4)},$$

para la cual no es posible despejar x en función de y:

```
(%i3) f(x):=(atan(1+exp(sin(x))))/(3-log(1+x^4));
```

(%o3) $$f\,(x) := \frac{atan\,(1 + exp\,(sin\,(x)))}{3 - log\,(1 + x^4)}$$

```
(%i4) solve(f(x)-y,x);
```

(%o4) $$[atan\left(e^{sin(x)}+1\right)=3\,y-log\left(x^4+1\right)\,y]$$

Para estos casos, resulta de mayor utilidad recurrir a otro método. Recordemos que una condición necesaria para la derivabilidad de la función inversa de una función $f:\mathbb{R}\to\mathbb{R}$ es la siguiente: si f es una función inyectiva, derivable en un punto x_0 y tal que su función inversa f^{-1} es derivable en $y_0=f(x_0)$ entonces, se cumple que $f'(x_0)\neq 0$. Además,

$$(f^{-1})'(y_0)=\frac{1}{f'(x_0)}.$$

La demostración es una simple aplicación de la regla de la cadena: por definición de función inversa, $f^{-1}\circ f=id$. Derivando y aplicando la regla de la cadena (teniendo en cuenta que la derivada de la identidad es la función lineal "multiplicar por la constante igual a 1"), resulta $f^{-1}(y_0)\cdot f(x_0)=1$, de donde se concluye la condición. Para el ejemplo que estamos considerando, supongamos que queremos hallar la derivada de la función inversa en el punto $\frac{\pi}{12}=f(0)$. Tendremos entonces que

$$g'\left(\frac{\pi}{12}\right)=\frac{1}{f'(0)}$$

y esto si se puede calcular directamente en Maxima:

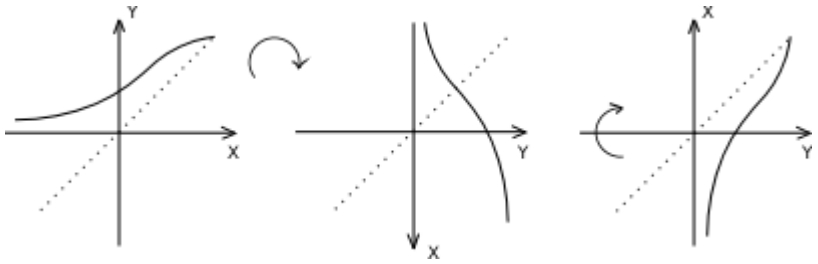
```
(%i5) gradef(g(y),g\'(y));
```

(%o5) $$g\,(y)$$

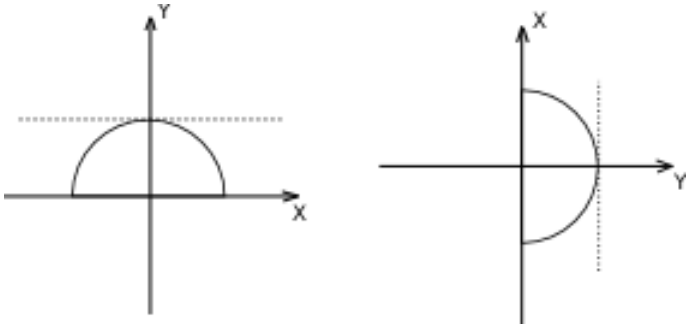
```
(%i6) g\'(%pi/12)=1/subst(x=0,diff(f(x),x));
```

(%o6) $$g'\left(\frac{\pi}{12}\right)=15$$

La función inversa f^{-1} de una función real de variable real inyectiva se obtiene, básicamente, intercambiando los ejes X e Y. Geométricamente esto puede hacerse mediante una rotación de 90 grados seguida de una reflexión, como en la figura:



El resultado final también puede verse como la gráfica de la función original reflejada con respecto a la bisectriz del primer cuadrante.
Es fácil entonces convencerse de la necesidad de la condición $f'(x_0) \neq 0$: en los puntos donde $f'(x_0) = 0$ toda la gráfica de la función está de un mismo lado de la recta tangente, toda por encima o toda por debajo. Al hacer el giro de 90 grados, toda la gráfica queda o bien toda a la derecha de la recta vertical o bien toda a la izquierda. En cualquier caso, lo que resulta no es una función, pues cada punto del nuevo eje de abcisas tiene dos imágenes:



7.2 Funciones inversas de funciones vectoriales. Cambios de coordenadas

En el caso multidimensional sabemos que el sustituto correcto de la derivada en un punto lo proporciona la diferencial en el punto, una aplicación lineal cuya matriz asociada (la matriz Jacobiana) está formada por las derivadas parciales de la función evaluadas en el punto en cuestión. Por eso, no es de extrañar que se tenga el resultado siguiente:
Sea $f : \mathbb{R}^n \to \mathbb{R}^n$ y $x_0 \in int(dom(f))$, con $y_0 = f(x_0)$. Supongamos que f es diferenciable en un entorno de x_0. Si se cumple que

$$det(J_{x_0}(f) \neq 0,$$

entonces, f admite una inversa local f^{-1} definida de un entorno V de y_0 en un entorno U de x_0, $f^{-1} : V \to U$, que es diferenciable en V y, además, cumple que

$$d_{y_0} f^{-1} = (d_{x_0} f)^{-1}.$$

Se dice entonces que f es un *difeomorfismo local* de U en V, o un *cambio de coordenadas*.
Veamos un ejemplo sencillo. Tenemos una aplicación $f : \mathbb{R}^2 \to \mathbb{R}^2$ dada por sus funciones componentes u, v, $f(x, y) = (u, v)$, donde $u = 2x + y$ y $v = x - y$. Nos preguntamos si esta f admite una inversa y, en caso afirmativo, alrededor de qué puntos. Para averiguarlo, calculamos su matriz Jacobiana, mediante el comando `jacobian` de que dispone Maxima. La sintaxis es `jacobian([f1,f2,...],[x1,x2,...])`, con `f1,f2,...` las funciones componentes de f y `x1,x2,...` las variables:

```
(%i7) jacobian([2*x+y,x-y],[x,y]);
```

$$0errors, 0warnings$$

(%o7)
$$\begin{pmatrix} 2 & 1 \\ 1 & -1 \end{pmatrix}$$

```
(%i8) determinant(%);
```

(%o8)
$$-3$$

Por tanto, la función $f(x,y) = (2x+y, x-y)$ cumple las hipótesis del teorema en todos los puntos de $\mathbb{R}^2$ y admitirá una inversa global que vendrá dada por una f^{-1} cuyas funciones componentes $f^{-1}(u,v) = (x(u,v), y(u,v))$ podemos determinar despejando el sistema $u = 2x + y$, $v = x - y$:

```
(%i9) algsys([2*x+y-u,x-y-v],[x,y]);
```

(%o9)
$$[[x = \frac{v+u}{3}, y = -\frac{2\,v-u}{3}]]$$

Consideremos ahora un ejemplo un poco más complicado. Supongamos que nuestra función $f : \mathbb{R}^2 \to \mathbb{R}^2$ ahora tiene funciones componentes $u = xy$, $v = x + y$. En este caso, su Jacobiana en un punto (x, y) será:

```
(%i10) jacobian([x*y,x+y],[x,y]);
```

(%o10)
$$\begin{pmatrix} y & x \\ 1 & 1 \end{pmatrix}$$

y su determinante:

```
(%i11) determinant(%);
```

(%o11)
$$y - x$$

Vemos pues que en los puntos donde el determinante se anula, esto es, a lo largo de la bisectriz del primer cuadrante, no será posible invertir la función. Además, esta recta singular divide al plano en dos partes en las que es de esperar que el comportamiento de f^{-1} sea distinto. De hecho, si tratamos de invertir la ecuaciones para obtener x, y en función de u, v resultan dos soluciones:

```
(%i12) algsys([x*y-u,x+y-v],[x,y]);
```

(%o12)

$$[[x=-\frac{\sqrt{v^2-4u}-v}{2},y=\frac{\sqrt{v^2-4u}+v}{2}]$$
$$[x=\frac{\sqrt{v^2-4u}+v}{2},y=-\frac{\sqrt{v^2-4u}-v}{2}]]$$

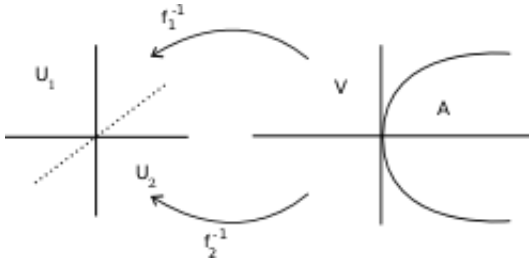
Así, dependiendo de la región del plano en que estemos calculando la inversa, la expresión de f^{-1} será una u otra. Para averiguar cuáles son esas regiones, observamos que la recta $x=y$ se transforma en la curva $u=\frac{v^2}{4}$, que es una parábola.
Si llamamos A al subconjunto abierto de $\mathbb{R}^2$ definido por la porción de plano que queda "dentro" de la parábola (esto es, los puntos (u,v) tales que $4u>v^2$) y V al abierto del plano definido por estar "fuera" de la parábola (esto es, los puntos (u,v) tales que $4u<v^2$), entonces resulta que f^{-1} sólo está definida en V. Y las dos inversas posibles transforman esta región V en dos regiones distintas del plano con coordenadas (x,y).
La primera inversa, que denominaremos f_1^{-1} y tomaremos como

$$f_1^{-1}(u,v)=(-\frac{1}{2}(\sqrt{(}v^2-4u)-v),\frac{1}{2}(\sqrt{(}v^2-4u)+v)),$$

transforma V en la región U_1 dada por $y>x$ (ya que $y-x=\sqrt{(}v^2-4u)>0$). Por su parte, la segunda función inversa

$$f_1^{-1}(u,v)=(\frac{1}{2}(\sqrt{(}v^2-4u)+v),-\frac{1}{2}(\sqrt{(}v^2-4u)-v))$$

transforma V en la región U_2 dada por $y<x$:



7.3 Funciones implícitas

Íntimamente relacionado con los resultados de la sección anterior, tenemos el teorema de la función implícita; éste, nos da condiciones bajo las cuales una ecuación $F(x,y)=0$ permite despejar una variable, digamos y, en función de la otra, para obtener la función *implícita* $y=\psi(x)$. Además, el teorema nos permite calcular las derivadas de esta función implícita, $y'=\frac{d\psi}{dx}$. Particularizado al caso de dos variables, el teorema puede enunciarse así: Sea una función

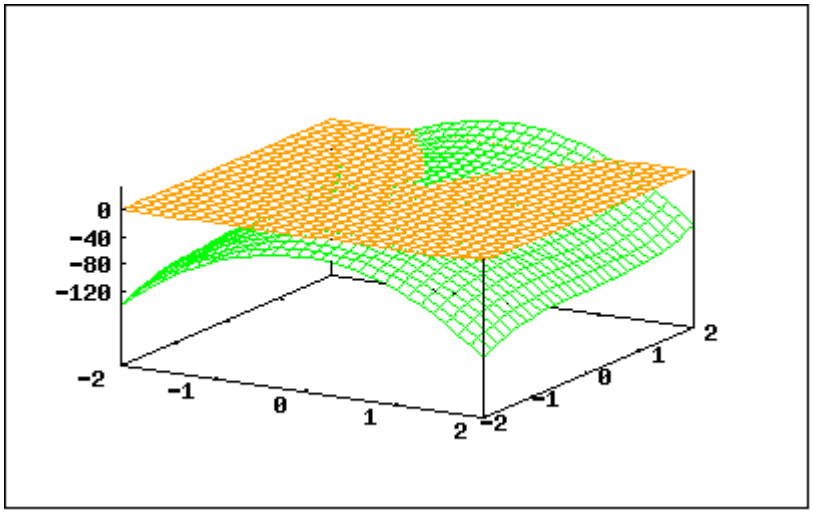
$F : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$ tal que sus derivadas parciales $D_1F = \frac{\partial F}{\partial x}$ y $D_2F = \frac{\partial F}{\partial y}$ son continuas en Ω. Sea $(x_0, y_0) \in \Omega$ tal que $F(x_0, y_0) = 0$. Si se cumple que $\frac{\partial F}{\partial y}(x_0, y_0) \neq 0$, entonces, existen unos entornos abiertos U de x_0 y V de y_0 tales que existe una única función $\psi : U \to V$ con las propiedades siguientes:

- $y_0 = \psi(x_0)$,
- $F(x, \psi(x)) = 0$, $\forall x \in U$,
- ψ es continuamente diferenciable y

$$\psi'(x) = -\frac{D_1F}{D_2F}(x, \psi(x))$$

Para ver el contenido geométrico del teorema, consideremos una superficie en $\mathbb{R}^3$, como por ejemplo la dada por la gráfica de la función $F(x, y) = -27x^2+4y^3$, y calculemos su intersección con el plano $z = 0$:

```
(%i13) load(draw)$
(%i14) wxdraw3d(color=green,
parametric_surface(u,v,-27*u^2+4*v^3,u,-2,2,v,-2,2),
color=orange,
parametric_surface(u,v,0,u,-2,2,v,-2,2),
surface_hide=true,
xtics=1,ytics=1,ztics=40);
```

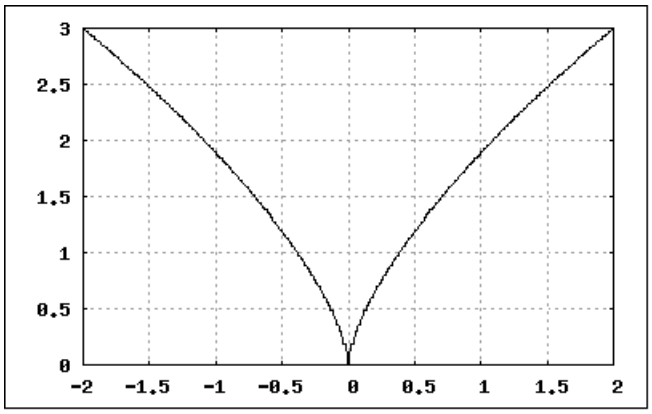


(%o14) $[gr3d(parametric\_surface, parametric\_surface)]$

Esa intersección es precisamente el lugar geométrico en $\mathbb{R}^3$ de los puntos $(x, y, 0)$ que verifican la ecuación $F(x, y) = -27x^2 + 4y^3 = 0$. Estos puntos,

determinan una curva en el plano xy, como se ve claramente en la figura anterior. De hecho, podemos pedirle a Maxima que nos represente esa curva. Para ello, utilizaremos el paquete `draw`, que ya hemos tratado. Esta vez, necesitaremos el comando `implicit` para reprentar una curva definida implícitamente. La sintaxis se explica por si misma:

```
(%i15) wxdraw2d(grid=true,
line_type=solid,
implicit(4*y^3=27*x^2,x,-2,2,y,0,3));
```



(%o15) $[gr2d\,(implicit)]$

En este ejemplo, se hace aparente el hecho de que la ecuación $F(x,y)=0$ define a y como función diferenciable de x en un entorno de cualquier punto que no sea el $(0,0)$, en la forma $y=\psi(x)$. Así, sobre la curva se cumple que $F(x,\psi(x))=0$.
De acuerdo con el teorema, dado un punto $(x_0,y_0))$ distinto del $(0,0)$, podemos calcular la derivada de esa función ψ en x_0. Usando Maxima, lo haríamos así:

```
(%i16) F(x,y):=-27*x^2+4*y^3;
```

(%o16) $F\,(x,y) := (-27)\,x^2 + 4\,y^3$

```
(%i17) depends(%psi,x);
```

(%o17) $[\psi\,(x)]$

```
(%i18) 'diff(%psi,x)=-diff(F(x,y),x)/diff(F(x,y),y);
```

(%o18) $$\frac{d}{d\,x}\,\psi = \frac{9\,x}{2\,y^2}$$

Naturalmente, se entiende que esta expresión se evalúa en (x_0, y_0). En otras palabras, esto debe leerse del modo siguiente: si (x_0, y_0) es un punto tal que $F(x_0, y_0) = 0$ y $\frac{\partial F}{\partial y}(x_0, y_0) \neq 0$, entonces está definida y como función implícita de x, $y = \psi(x)$, y su derivada en el punto x_0 vale

$$\frac{d\psi}{dx}(x_0) = \frac{9x_0}{2y_0^2}$$

. Una observación: hemos utilizado `%psi` como nombre para la función implícita. Esto muestra cómo en wxMaxima es posible utilizar letras griegas para denotar variables o funciones, pero esto requiere que el sistema tenga instaladas fuentes que admitan estos caracteres. Suponiendo que sea el caso, para habilitarlas hay que ir al menú principal de wxMaxima, Edit - Configure -Style y ahí marcar "Use greek font".

Veamos, para finalizar, otra aplicación típica del teorema de la función implícita. Supongamos que nos dan una curva plana en forma del conjunto de puntos solución a la ecuación $y^2 + 5x = xe^{x(y-2)}$ y nos piden calcular la ecuación de la recta tangente por el punto $(-1, 2)$. Procederíamos así: en primer lugar, comprobamos que el punto $(x_0, y_0) = (-1, 2)$ está sobre la curva:

```
(%i19) subst([x=-1,y=2],y^2+5*x=x*exp(x*(y-2)));
```

(%o19) $$-1 = -1$$

Una vez que sabemos que es así, comprobamos que se cumplen las hipótesis del teorema y podemos expresar y como función de x, $y = g(x)$:

```
(%i20) G(x,y):=y^2+5*x-x*exp(x*(y-2));
```

(%o20) $$G\,(x, y) := y^2 + 5\,x + (-x)\;exp\,(x\;(y-2))$$

```
(%i21) 'diff(G,y)=subst([x=-1,y=2],diff(G(x,y),y));
```

(%o21) $$\frac{d}{d\,y}\,G = 3$$

Ahora, ya podemos calcular la derivada de g:

```
(%i22) depends(g,x);
```

(%o22) $[g(x)]$

```
(%i23) 'diff(g,x)=-diff(G(x,y),x)/diff(G(x,y),y);
```

(%o23) $$\frac{d}{dx}g = \frac{x\,(y-2)\,e^{x\,(y-2)} + e^{x\,(y-2)} - 5}{2\,y - x^2\,e^{x\,(y-2)}}$$

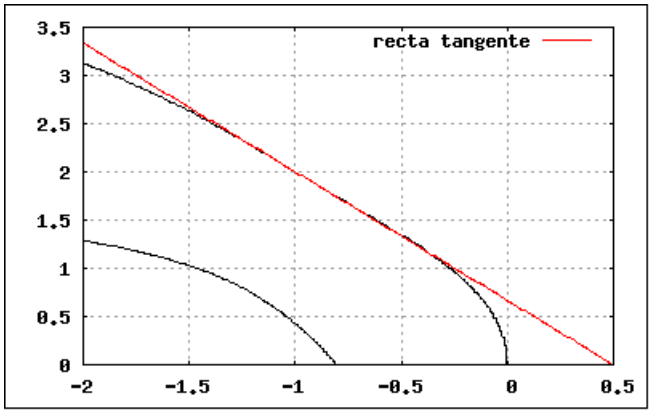
Y evaluarla en el punto $(x_0, y_0) = (-1, 2)$:

```
(%i24) g\'(-1)=subst([x=-1,y=2],-diff(G(x,y),x)/diff(G(x,y),y));
```

(%o24) $$g'(-1) = -\frac{4}{3}$$

Tenemos los siguientes datos sobre la recta: pasa por el punto $(-1, 2)$ y tiene pendiente $-4/3$. Con esto, resulta inmediato representarla, y lo haremos simultáneamente con la curva. Aprovechamos para mencionar otra posibilidad que ofrece el paquete `draw`, la de poder etiquetar las gráficas con una cadena de texto, que debe pasarse como argumento `key` entre comillas, como en el ejemplo:

```
(%i25) wxdraw2d(grid=true,
line_type=solid,
implicit(y^2+5*x=x*exp(x*(y-2)),x,-2,0.5,y,0,3.5),
color=red,
key="recta tangente",
implicit(y-2=-4/3*(x+1),x,-2,0.5,y,0,3.5));
```



(%o25) $[gr2d(implicit, implicit)]$

7.4 Ejercicios

1. Analizar la invertibilidad de la función dada por

$$y = f(x) = 1 - \frac{x^2}{x^4 + 1}.$$

2. Estudiar el cambio de coordenadas en $\mathbb{R}^2$ dado por la función $F : \mathbb{R}^2 \to \mathbb{R}^2$ cuyas componentes son

$$\begin{cases} u = x \\ v = x^2 + y \end{cases}$$

3. Utilizar el teorema de la función implícita para hallar la derivada de $y = \psi(x)$ siendo:

   a) $xy = 1$
   b) $x - y + 3xy = 2$
   c) $y^6 - x^5 = 0$.

4. Dada la curva plana cuyos puntos son solución a la ecuación

$$2x^2 + xy + y^2 = 8,$$

   a) Calcular la ecuación de la recta tangente a la curva en el punto $(2, 0)$.
   b) ¿Qué puntos de la curva tienen la tangente en ellos horizontal?.

5. Estudiar si la ecuación

$$(x + 2y + xy)e^{x^2 - y} = 4$$

   define a x como función implícita de y en un entorno del punto $(1, 1)$.

6. Explicar por qué la ecuación $e^{xy} - 1 = 0$ no define, alrededor del punto $(0, 0)$, a y como función de x, pero si lo hace alrededor del punto $(1, 0)$.

7. Dada la ecuación

$$2xe^y + x^2 + y^3 = -1,$$

   a) Probar que define a y como función de x alrededor del punto $(-1, 0)$.
   b) Comprobar que en $x = -1$ la función $y = f(x)$ tiene un mínimo local.

8. Consideremos la curva plana

$$y = xy^2.$$

   a) Comprobar que en un entorno del punto $(1, 1)$ la ecuación de la curva define a $y = g(x)$ como función implícita de x. Calcular $g(1)$.
   b) Calcular $g'(x)$.
   c) Determinar el desarrollo en serie de $y = g(x)$ a orden 2 alrededor de $x = 1$.

GNU Free Documentation License

Version 1.2, November 2002

51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. Applicability and definitions

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "**Document**", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "**you**". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "**Modified Version**" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "**Secondary Section**" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related

matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "**Invariant Sections**" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "**Cover Texts**" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "**Transparent**" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "**Opaque**".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "**Title Page**" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "**Entitled XYZ**" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "**Acknowledgements**", "**Dedications**", "**Endorsements**", or "**History**".) To "**Preserve the Title**" of such a section when you modify the Document means that it remains a section "Entitled XYZ"

according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. Verbatim copying

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. Copying in quantity

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. Modifications

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. Combining documents

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. Collections of documents

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. Aggregation with independent works

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. Translation

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. Termination

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. Future revisions of this license

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.