PRACTICA 2. REPRESENTACIONES GRÁFICAS EN 2D y 3D. CÁLCULO DIFERENCIAL E INTEGRAL INTEGRAL EN UNA VARIABLE.

1 INTRODUCCIÓN.

El alumno debe ir realizando todos los EJEMPLOS que aparecen en esta práctica e ir anotando los resultados en la hoja que se le ha entregado. Además de estos ejemplos, hay planteados unos EJERCICIOS, que, a diferencia de lo que ocurre con los EJEMPLOS, no vienen desarrollados en el lenguaje Maxima, sino que tendrá que ser el alumno el que tenga que introducirlos para obtener el resultado.

Por ejemplo, si en la práctica aparece

```
--> 3+5;
```

el alumno tendrá que situarse en la línea anterior, y darle a INTRO del teclado numérico para obtener el resultado.

Sin embargo, si aparece

EJERCICIO 1: Resolver la siguiente operación:
3+5

tendrá que ser el alumno el que introduzca el resultado (por ejemplo a continuación de esta línea (pinchar con el ratón debajo de esta línea y darle a ENTER para que aparezca la flecha en rojo donde introducir la sentencia:

2 GRÁFICOS EN EL PLANO CON plot2d.

2.1 Coordenadas cartesianas

El comando que se utiliza para representar la gráfica de una función de una variable real es plot2d que actúa, como mínimo, con dos parámetros: la función (o lista de funciones a representar), y el intervalo de valores para la variable x. Al comando plot2d se puede acceder también a través del menú Gráficos->Gráficos 2D

```
--> /*EJEMPLO 1*/
    plot2d(sin(2*x),[x,-2*%pi,2*%pi]);
```

```
--> /*EJEMPLO 2*/
    plot2d([x^2,sqrt(2*x)],[x,-2,2]);
```

Cuando pulsamos en el menú Gráficos->Gráficos 2D , aparece una ventana de diálogo con varios campos que podemos completar o modificar:
a) Expresión(es). La función o funciones que queramos dibujar. Por defecto, wxMaxima rellena este espacio con % para referirse a la salida anterior.
b) Variable x. Aquí establecemos el intervalo de la variable x donde queramos representar la función.
c) Variable y. Ídem para acotar el recorrido de los valores de la imagen.
d) Graduaciones. Nos permite regular el número de puntos en los que el programa evalúa una función para su representación en polares. Veremos ejemplos en la sección siguiente.

e) Formato. Maxima realiza por defecto la gráfica con un programa auxiliar. Si seleccionamos "en línea", dicho programa auxiliar es wxMaxima y obtendremos la gráfica en una ventana alineada con la salida correspondiente. Hay dos opciones más y ambas abren una ventana externa para dibujar la gráfica requerida: "gnuplot" es la opción por defecto que utiliza el programa Gnuplot para realizar la representación; también está disponible la opción "openmath" que utiliza el programa XMaxima. Prueba las diferentes opciones y decide cuál te gusta más.
f) Opciones. Aquí podemos seleccionar algunas opciones para que, por ejemplo, dibuje los ejes de coordenadas ("set zeroaxis;"); dibuje los ejes de coordenadas, de forma que cada unidad en el eje Y sea igual que el eje X ("set size ratio 1; set zeroaxis;"); dibuje una cuadrícula ("set grid;") o dibuje una gráfica en coordenadas polares ("set polar; set zeroaxis;"). Esta última opción la comentamos más adelante.
g) Gráfico al archivo. Guarda el gráfico en un archivo con formato Postscript.
Evidentemente, estas no son todas las posibles opciones. La cantidad de posibilidades que tiene Gnuplot es inmensa.
Observación. El prefijo “wx” añadido a plot2d o a cualquiera del resto de las órdenes que veremos en esta práctica (plot3d, draw2d, draw3d) hace que wxMaxima pase automáticamente a mostrar los gráficos en la misma ventana y no en una ventana separada. Es lo mismo que seleccionar en línea. Por ejemplo,

```
-->  /*EJEMPLO 3*/
     wxplot2d(sin(2*x),[x,-2*%pi,2*%pi]);
```

```
-->  /*EJEMPLO 4*/
     plot2d(x/(x^2-4),[x,-6,6],[y,-6,6],
     [gnuplot_preamble, "set zeroaxis;"])$
```

```
-->  /*EJEMPLO 5*/
     plot2d(x/(x^2-4),[x,-6,6],[y,-6,6],
     [gnuplot_preamble, "set size ratio 1; set zeroaxis;"])$
```

```
-->  /*EJEMPLO 6*/
     plot2d(x/(x^2-4),[x,-6,6],[y,-6,6],
     [gnuplot_preamble, "set grid;"])$
```

2.2 Coordenadas polares

Al representar una curva en coordenadas polares estamos escribiendo la longitud del vector como una función que depende del ángulo. En otras palabras, para cada ángulo fijo decimos cuál es el módulo del vector. El ejemplo más sencillo de función que a cualquiera se nos viene a la cabeza son las funciones constantes: La función f :[0,2pi]->R, f()=1 tiene como imagen aquellos vectores que tienen módulo 1 y argumento entre 0 y 2pi. Para ello, tenemos que seleccionar
"set polar; set zeroaxis;" en el campo Opciones de Gráficos 2D:

```
-->  /*EJEMPLO 7*/
     plot2d([1], [ph,0,2*%pi],
     [plot_format, gnuplot],
     [gnuplot_preamble, "set polar; set zeroaxis;"],[x,-1,1])$
```

y si queremos que la gráfica sea proporcionada, hacemos

```
-->  /*EJEMPLO 8*/
     plot2d([1], [ph,0,2*%pi],
     [plot_format, gnuplot],
     [gnuplot_preamble, "set polar;set size ratio 1; set zeroaxis;"],
     [x,-1,1])$
```

Otra gráfica viene dada por

```
-->  /*EJEMPLO 9*/
     plot2d(ph,[ph,0,4*%pi],
     [gnuplot_preamble, "set polar; set zeroaxis;"],[x,-15,15])$
```

Observamos que la hélice resultante no es nada “suave”. Para conseguir el efecto visual de una línea curva como es esta hélice, añadimos el parámetro nticks. Por defecto, para dibujar una gráfica en paramétricas el programa evalúa en 10 puntos. Para aumentar este número de puntos, aumentamos dicho parámetro, por ejemplo nticks=30, o bien, podemos regularlo desde el campo Graduaciones dentro de de Gráficos 2D.

```
-->  /*EJEMPLO 10*/
     plot2d(ph,[ph,0,4*%pi],
     [gnuplot_preamble, "set polar; set zeroaxis;"], [nticks,30],
     [x,-15,15])$
```

Si representamos la función
r(theta)=exp(cos(theta))-2cos(4theta)+sin(theta/12)^5
obtenemos algo parecido a una mariposa:

```
-->  /*EJEMPLO 11*/
     r(ph):=exp(cos(ph))-2*cos(4*ph)+sin(ph/12)^5$
     plot2d(r(ph),[ph,0,2*%pi],
     [gnuplot_preamble, "set polar; set zeroaxis;"],[x,-5,5])$
```

2.3 Coordenadas paramétricas

El programa wxMaxima nos permite también representar curvas en forma paramétrica, es decir, curvas definidas como (x(t),y(t)), donde el parámetro t varía en un determinado intervalo compacto [a,b].
Para ello, dentro del comando plot2d añadimos “parametric” de la forma siguiente: plot2d([parametric,x(t),y(t),[t,a,b]])
Para acceder a esta opción de la función plot2d podemos hacerlo a través del botón Especial que aparece en la parte superior derecha de la ventana de diálogo Gráficos 2D.
Para terminar, aquí tienes algunas curvas planas interesantes:
Astroide: Es la curva trazada por un punto fijo de un círculo de radio r que rueda sin deslizar dentro de otro círculo fijo de radio 4r.
Sus ecuaciones paramétricas son:

```
-->  /*EJEMPLO 12*/
     plot2d([parametric,cos(t)^3,sin(t)^3,[t,0,2*%pi],[nticks,50]])
```

Cardioide: Es la curva trazada por un punto fijo de un círculo de radio r que rueda sin deslizar alrededor de otro círculo fijo del mismo radio.
Sus ecuaciones paramétricas
son:

```
-->  /*EJEMPLO 13*/
     plot2d([parametric,(1+cos(t))*cos(t),(1+cos(t))*sin(t),
     [t,0,2*%pi],[nticks,50]])
```

Lemniscata de Bernoulli:Es el lugar geométrico de los puntos P del plano, cuyo producto de distancias a dos puntos fijos F1 y F2, llamados focos, verifica la igualdad abs(P-F1)*abs(P-F2)=1/4*abs(F1-F2)^2
En coordenadas cartesianas esta curva viene dada por la ecuación (x^2+y^2)^2=x^2-y^2. Aquí tienes sus ecuaciones paramétricas:

```
-->  /*EJEMPLO 14*/
     plot2d([parametric, (cos(t))/(1+sin(t)^2),
     (cos(t)*sin(t))/(1+sin(t)^2), [t,0,2*%pi],[nticks,70]])
```

Espiral equiangular: También llamada espiral logarítmica. Es aquella espiral en la que el radio vector corta a la curva en un ángulo constante alfa. Sus ecuaciones paramétricas son:

```
-->  /*EJEMPLO 15*/
     %alpha:%pi/2-0.2$
     plot2d([parametric,exp(t*cot(%alpha))*cos(t),
     exp(t*cot(%alpha))*sin(t),[t,0,4*%pi],[nticks,90]])
```

Cicloide: También conocida como tautocrona o braquistocrona. Es la curva que describiría un punto de una circunferencia que avanza girando sin deslizar. Sus ecuaciones paramétricas son:

```
-->  /*EJEMPLO 16*/
     plot2d([parametric,t-sin(t),1-cos(t),[t,0,6*%pi],[nticks,90]])
```

3 GRÁFICOS EN 3D.

Con Maxima se pueden representar funciones de dos variables de forma similar a como hemos representado funciones de una. La principal diferencia es que vamos a utilizar el comando plot3d en lugar de plot2d, pero igual que en el caso anterior, son obligatorios la función o funciones y el dominio que tiene que ser de la forma [a,b]x[c,d]:
plot3d(f(x,y),[x,a,b],[y,c,d]) gráfica de f(x,y) en [a,b]x[c,d]

```
 -->  /*EJEMPLO 17*/
      plot3d(cos(x*y),[x,-3,3],[y,-3,3]);
```

La gráfica que acabas de obtener se puede girar sin más que pinchar con el ratón sobre ella y deslizar el cursor.
plot3d también permite trazar la gráfica de una superficie definida en forma paramétrica; sin embargo, no permite gráficas de curvas en R3: Para ello usaremos
plot3d([x(u,v),y(u,v),z(u,v)],[u,umin,umax],[v,vmin,vmax])
Así, si queremos representar la esfera unitaria definida paramétricamente mediante (u,v)->(sen u sen v,cos u sen v,cos v), con u entre 0 y 2pi, y v entre -pi/2 y pi/2:

```
 -->  /*EJEMPLO 18*/
      plot3d([cos(u)*cos(v),sin(u)*cos(v),sin(v)],
      [u,0,2* %pi],[v,- %pi/2, %pi/2]);
```

Al comando plot3d se puede acceder a través del menú Gráficos->Gráficos 3D. Después de esto aparece una ventana con varios campos para rellenar:

a) Expresión. La función o funciones que vayamos a dibujar.
b) Variable. Hay dos campos para indicar el dominio de las dos variables.
c) Cuadrícula. Indica cuántas valores se toman de cada variable para representar la función. Cuanto mayor sea, más suave será la representación a costa de aumentar la cantidad de cálculos.
d) Formato. Igual que en plot2d, permite escoger qué programa se utiliza para representar la función. Se puede girar la gráfica en todos ellos salvo si escoges “en línea”.
e) Opciones. Las comentamos a continuación.
f) Gráfico al archivo. Permite elegir un fichero donde se guardará la gráfica.
Quizá la mejor manera de ver el efecto de las opciones es repetir el dibujo anterior. La primera de ellas es "set pm3d at b" que dibuja la superficie usando una malla y en la base añade curvas de nivel (como si estuviéramos mirando la gráfica desde arriba):

```
 -->  /*EJEMPLO 19*/
      plot3d(cos(x*y), [x,-5,5], [y,-5,5], [plot_format,gnuplot],
      [gnuplot_preamble, "set pm3d at b"])$
```

La segunda hace varias cosas, "set pm3d at s" nos dibuja la superficie coloreada, "unset surf" elimina la malla y "unset colorbox" elimina la barra que teníamos en la derecha con la explicación de los colores y su relación con la altura (el valor de la función):

```
--> /*EJEMPLO 20*/
    plot3d(cos(x*y), [x,-5,5], [y,-5,5], [plot_format,gnuplot],
    [gnuplot_preamble, "set pm3d at s; unset surf;
    unset colorbox"])$
```

La tercera, "set pm3d map", nos dibuja un mapa de curvas de nivel (gráfico de densidad) con alguna ayuda adicional dada por el color: Así, si queremos representar solamente el gráfico de densidad para esta superficie, haremos:

```
--> /*EJEMPLO 21*/
    plot3d(cos(x*y), [x,-5,5], [y,-5,5],
    [gnuplot_preamble,"set pm3d map"]);
```

La cuarta, "set hidden3d", sólo muestra la parte de la superficie que sería visible desde nuestro punto de vista. En otras palabras, hace la superficie sólida y no transparente:

```
--> /*EJEMPLO 22*/
    plot3d(cos(x*y), [x,-5,5], [y,-5,5], [plot_format,gnuplot],
    [gnuplot_preamble, "set hidden3d"])$
```

En el dibujo anterior (en el papel) es posible que no aprecie bien. A simple vista parece el mismo dibujo que teníamos dos salidas antes. Observa bien: hay una pequeña diferencia. El uso de pm3d hace que se coloree el dibujo, pero cuando decimos que no se muestra la parte no visible de la figura nos estamos refiriendo a la malla. Quizá es mejor dibujar la malla y el manto de colores por separado para que se vea la diferencia. Esta opción no viene disponible por defecto en wxMaxima. Ten en cuenta que las opciones que tiene Gnuplot son casi infinitas y sólo estamos comentando algunas.

```
--> /*EJEMPLO 23*/
    plot3d(x^2-y^2, [x,-5,5], [y,-5,5], [plot_format,gnuplot],
    [gnuplot_preamble, "set pm3d at b; set hidden3d"])$
```

```
--> /*EJEMPLO 24*/
    plot3d(x^2-y^2, [x,-5,5], [y,-5,5], [plot_format,gnuplot],
    [gnuplot_preamble, "set pm3d at b"])$
```

La quinta y la sexta opciones nos permiten dibujar en coordenadas esféricas o cilíndricas. Ya veremos ejemplos más adelante.
Si queremos representar solamente un gráfico de contorno (curvas de nivel), es posible mediante contour_plot, como vemos a continuación:

```
--> /*EJEMPLO 25*/
    contour_plot(cos(x*y), [x,-5,5], [y,-5,5])$
```

4 GRÁFICOS CON draw.

El paquete draw se distribuye conjuntamente con Maxima y constituye una interfaz que comunica de manera muy eficiente Maxima con Gnuplot. Este paquete incorpora una considerable variedad de funciones y opciones que permiten obtener la representación de un amplio número de objetos gráficos bidimensionales y tridimensionales. En este caso nos vamos a centrar solamente en GRAFICOS TRIDIMENSIONALES, ya que con draw vamos a poder realizar gráficas que no podemos realizar con plot3g, como por ejemplo, representar curvas en R3. No obstante, invitamos al lector a que profundice en este paquete (a menudo mucho más sencillo e intuitivo que plot) para ver otras opciones. Toda la información al respecto se puede ver fácilmente en el manual que se encuentra en:
http://www.ugr.es/~dpto_am/docencia/Apuntes/Practicas_con_Maxima.pdf
Para poder utilizar el paquete draw es preciso cargarlo en la memoria, y para ello se utiliza la función load(draw).
Las funciones draw, draw2d y draw3d devuelven las salidas gráficas en una ventana de Gnuplot, aparte de la ventana de trabajo actual.
No obstante el entorno gráfico wxMaxima permite utilizar las funciones wxdraw, wxdraw2d y wxdraw3d que sí devuelven las salidas en el cuaderno de trabajo actual. Debe tenerse presente que al utilizar la función wxdraw3d, el punto de vista de la gráfica obtenida no puede ser cambiado en tiempo real.
En esta práctica se van a mostrar los resultados mediante las función draw3d, pero todos los ejemplos mostrados pueden ser ejecutados, sin ningún problema, con la función wxdraw3d (lo mismo ocurre con draw y draw2d, como puede verse en el manual anteriormente citado).

Principales objetos gráficos tridimensionales incorporados en draw:
- points([x1,x2,...];[y1,y2,,,,],[z1,z2,...]) representa puntos en R3
- vector([p1,p2,p3],[v1,v2,v3]) nos da el vector (v1,v2,v3) con punto de aplicación (p1,p2,p3)
- explicit(f(x,y),x,xmin,xmax,y,ymin,ymax) representa a una función explícita f en el dominio dado.
- implicit(E(x,y,z),x,xmin,xmax,y,ymin,ymax,z,zmin,zmax) representa a una ecuación implícita E en el dominio dado
- parametric(fx,fy,fz,t,tmin,tmax) curva paramétrica tridimensional, cuyo parámetro t satisface tmin<t<tmax
- cylindrical(r(z,theta),z,zmin,zmax,theta,thetamin,thetamax), nos da una función cilíndrica r donde los parámetros varían en los intervalos que se indican
- spherical(r(fi;theta),fi,fimin,fimax,theta,thetamin,thetamax) nos da una función esférica r donde los parámetros varían en los intervalos que se indican

Veamos ejemplos de todos éstos comandos (y otros):

Aquí se muestra la gráca del vector cuyo punto de aplicación es el origen y cuya parte vectorial es (1,1,1): (Primero tenemos que llamar al paquete draw)

```
-->  load(draw)$
```

```
-->  /*EJEMPLO 26*/
     draw3d(vector([0,0,0],[1,1,1]));
```

Esto muestra la gráfica de la función f(x,y)=sen x + sen(xy), en [0,2pi]x[0,2pi]:

```
 -->  /*EJEMPLO 27*/
      draw3d(explicit(sin(x)+sin(x*y),x,0,2*%pi,y,0,2*%pi));
```

Aquí se muestra la gráfica de una superficie generada a partir de una ecuación implícita (se indica la ecuación y donde varían las 3 variables x, y, z):

```
 -->  /*EJEMPLO 28*/
      draw3d(implicit((sqrt(x^2+y^2)-4)^2+z^2=4,x,-6,6,
      y,-6,6,z,-2,2) );
```

Esta es la gráfica de la curva dada por la parametrización t->(cos t,sen t,t/8), con 0<t<4pi (estas curvas en R3 no podemos representarlas mediante plot3d):

```
 -->  /*EJEMPLO 29*/
      draw3d(parametric(cos(t),sin(t),t/8,t,0,4*%pi));
```

Aquí se muestra la gráfica de la superficie definida por la parametrización (u,v)->((2+cos v)cos u,(2+cos v)sin u,sin v), con 0<u<2pi y 0<v<2pi:

```
 -->  /*EJEMPLO 30*/
      draw3d(parametric_surface((2+cos(v))*cos(u),
      (2+cos(v))*sin(u),sin(v),u,0,2*%pi,v,0,2*%pi));
```

Esta es la gráfica de la superficie definida en coordenadas cilíndricas mediante (z,t)->cos z, con -2<z<2 y 0<t<2pi:

```
 -->  /*EJEMPLO 31*/
      draw3d(cylindrical(cos(z),z,-2,2,t,0,2*%pi));
```

He aquí la gráfica de la superficie definida en coordenadas esféricas mediante (a,t)->1, con 0<a<2pi y 0<t<pi:

```
 -->  /*EJEMPLO 32*/
      draw3d(spherical(1,a,0,2*%pi,t,0,%pi));
```

A continuación incluimos otros ejemplos donde aparecen determinadas opciones que podemos realizar:

Esto define una curva, algunos puntos sobre ésta y algunos vectores tangentes a la misma:

```
--> /*EJEMPLO 33*/
a(t):=[cos(t),sin(t),t/8]$
define("a'"(t),diff(a(t),t))$
t0:create_list(i*%pi/4,i,0,16)$
T:create_list( vector(a(i),"a'"(i)),i,t0)$
P:map(a,t0)$
objetos:([nticks=100,color=red,
apply(parametric,append(a(t),[t,0,4*%pi]))
color=blue,unit_vectors=true,T,
color=green,point_type=7,points(P),
user_preamble="set size ratio 1",
title="Campo vectorial tangente",
xyplane=0,axis_3d=false,
xtics=false,ytics=false,ztics=false,
xaxis=true,yaxis=true,zaxis=true,
xlabel="x",ylabel="y",zlabel="z"])$
draw3d(objetos);
```

La superficie implícita definida anteriormente pero con otras opciones:

```
--> /*EJEMPLO 34*/
draw3d(x_voxel=17,y_voxel=17,z_voxel=15,
palette=[12,5,27],enhanced3d=true,
implicit((sqrt(x^2+y^2)-4)^2+z^2=4,x,-6,6,
y,-6,6,z,-2,2));
```

5 CÁLCULO DIFERENCIAL EN UNA VARIABLE.

5.1 SUCESIONES DE NÚMEROS REALES

En Maxima, las sucesiones se definen a través de su término general o bien mediante una expresión de recurrencia. En cualquier caso, se utilizará el operador :=, introduciendo la variable índice entre corchetes [].

Por ejemplo, podemos introducir la sucesión de término general a[n]:=1/(2*n-1); podemos calcular su primer término o los diez primeros para lo cual tendremos que hacer sucesivamente:

```
--> /*EJEMPLO 35*/
a[n]:=1/(2*n-1);
a[1];
makelist(a[k],k,1,10);
```

También podemos definir sucesiones por recurrencia. En este caso deberemos, además, utilizar el operador : para asignar valores a los primeros elementos, como hacemos a continuación

```
--> /*EJEMPLO 36*/
b[1]:1;
b[n]:=2*b[n-1];
makelist(b[k],k,1,10);
```

Para Calcular límites usamos la función limit, que está disponible en el menú Análisis->Calcular límite... de wxMaxima. Como parámetros de esta función, tenemos que teclear la expresión de una sucesión, el nombre de la variable respecto de la cual deseamos calcular el límite y el valor hacia el que ésta tiende. En wxMaxima existen dos variables especiales, inf y minf, que representan respectivamente a +infinito y -infinito.

```
-->  /*EJEMPLO 37*/
     c[n]:=(n-sqrt(n^2-4))/(n+1);
     limit(c[n],n,inf);
```

EJERCICIO 1: Calcular los siguientes límites:
a) limit(n/sqrt(n),n,inf).
b) limit(1/(sqrt(n)-sqrt(n+1)),n,inf).
c) limit(((n-2)/(n-3))^(2*n),n,inf).
d) limit((-1)^n,n,inf).
e) limit(n*(-1)^n,n,inf).

Como se puede observar, ante el cálculo del límite de (-1)^n, Maxima devuelve la palabra clave "ind", que se puede interpretar como indefinido pero acotado, mientras que ante el límite n*(-1)^n responde "und" (de undefined) para indicar que el límite es indefinido y no necesariamente acotado:

```
-->  /*EJEMPLO 38*/
     limit((-1)^n,n,inf);
     limit(n*(-1)^n,n,inf);
```

5.2 ESTUDIO DE UNA FUNCIÓN REAL DE VARIABLE REAL

Veamos con un ejemplo el estudio completo de una función real de variable real. Para ello, consideremos la función
f(x)=1/(1+x^2), si x<=1; f(x)=1+log(x), si x>1
y veremos como podemos hacerle un estudio local completo:

```
-->  /*EJEMPLO 39*/
     /*EJEMPLO 39.1: En primer lugar veremos como estudiar su
     dominio, puntos de corte y asíntotas*/
     g(x):=1/(1+x^2);
     h(x):=1+log(x);
```

Sabemos que el dominio de g(x) es todo R, puesto que el denominador no se anula nunca, aunque si hubiese duda resolvemos la ecuación que aparece en su denominador

```
-->  solve(1+x^2=0);
```

Por otra parte, la función h(x) solamente está bien definida para aquellos valores de x para los que tiene sentido log(x), es decir, el dominio de h está formado por todos los números reales positivos. Por tanto el dominio de f(x) son todos los números reales.

```
--> /*EJEMPLO 39.1: continuación*/
    /*Para estudiar los puntos de corte con el eje de las abscisas,
    planteamos las siguientes ecuaciones*/
    solve(g(x)=0,x);
    solve(h(x)=0,x);
```

Para la ecuación g(x) = 0 no existen soluciones lo que significa que no hay puntos de corte con el eje x para x<=1. En el caso de la ecuación h(x) =0, obtenemos la solución x = e^(-1), pero este valor está fuera del intervalo de definición (donde se aplica h(x) en la función f(x)), con lo que concluimos que no existe ningún punto de corte de la gráfica de la función f(x) con el eje x.

En cuanto a posibles puntos de corte con el eje vertical, cuando x = 0 nuestra función toma el valor f(0) = g(0) = 1:

```
--> g(0);
```

En definitiva, el único punto de corte de f(x) con los ejes es el (0,1).

Para estudiar las asíntotas verticales, tendríamos que analizar si existe algún punto a en el que el límite de f(x) por la derecha o por la izquierda sea + ó - infinito, lo que ocurre típicamente en funciones racionales en las que se anula el denominador, en funciones logarítmicas, etc... En nuestro caso, la función g(x) no tiene ninguna asíntota vertical, porque su denominador es siempre distinto de cero. La función h(x) tendría una asíntota vertical en x=0, debido al logaritmo y teniendo en cuenta el dominio del logaritmo, sólo tiene sentido calcular el límite cuando x tiende a cero por la derecha:

```
--> /*EJEMPLO 39.1: continuación*/
    limit(h(x),x,0,plus);
```

Pero esto no afecta a f(x), ya que en los alrededores de x = 0 no toma los valores de h(x), sino de g(x), con lo que f(x) no tiene ninguna asíntota vertical.

Con respecto a asíntotas horizontales, tendremos que estudiar límites de f(x)cuando x->-infinito (en cuyo caso f(x)=g(x)) y cuando x->+inf (en cuyo caso f(x)=h(x)),

```
--> /*EJEMPLO 39.1: continuación*/
    limit(g(x),x,minf);
    limit(h(x),x,inf);
```

Por lo tanto, podemos concluir que f(x) no tiene ninguna asíntota vertical y sí tiene una asíntota horizontal (la recta y = 0) cuando x->-infinito.

También podemos estudiar la continuidad y derivabilidad de f(x):
Las funciones g(x) y h(x) son continuas dentro de sus respectivos dominios, por lo tanto f es continua salvo, eventualmente, en x = 1, que es el punto que divide las regiones donde f(x) toma los valores de g(x) y de h(x). Pero los límites laterales de f(x) en este punto son distintos, pues:

```
-->  /*EJEMPLO 39.2: Estudio de la continuidad y derivabilidad*/
     limit(g(x),x,1,minus);
     limit(h(x),x,1,plus);
```

en consecuencia, f(x) no es continua en x = 1 (tendrá una discontinuidad de salto finito).

Además, f(x) no es derivable en x = 1 (por no ser continua) pero sí en el resto de R (pues tanto g(x) como h(x) lo son para valores de x que estén dentro de sus respectivos dominios). Puesto que

```
-->  diff(g(x),x);
     diff(h(x),x);
```

Por tanto la función derivada de f(x) es: f'(x)=-2x/(x^2+1)^2, si x<1; f'(x)=1/x, si x>1

NOTA: En este último ejemplo hemos aprendido como se calcula la derivada de una función. Ésto también puede hacerse usando los comandos que nos encontramos en Análisis->Derivar...

También podemos estudiar el crecimiento y/o la existencia de extremos relativos:

El crecimiento de f(x) depende del signo de la derivada, f'(x).
Cuando x > 1, obviamente, 1/x > 0 y por lo tanto, f'(x) > 0.
Aunque es un caso tan sencillo que no merece la pena recurrir al ordenador, puede servir como ilustración la forma en que se podría comprobar lo anterior, utilizando los comandos assume e is:

```
-->  /*EJEMPLO 39.3: Estudio de la monotonía y extremos*/
     assume(x>1);
     is(1/x>0);
```

cuando x < 1, el asunto es diferente, pues el signo de f'(x) dependerá de la expresión -2x/(x^2+1)^2, que depende del signo de -2x (puesto que el denominador es siempre estrictamente positivo), siendo, negativo cuando x > 0 y positivo en caso contrario.
Por lo tanto f(x) es creciente en (-inf,0) y (1,+inf) y decreciente en (0,1).

Aunque en casos tan sencillos como este no merezca la pena, el signo de -2x/(x^2+1)^2 se podría haber estudiado utilizando el ordenador. Aunque ante una pregunta inicial, Maxima se muestra incapaz de ofrecer una respuesta:

```
-->  forget(x>1)$
     assume(x<1)$
     expresion:diff(g(x),x);
     is(expresion>0);
```

NOTA: Es necesario utilizar el comando forget, que elimina la restricción que fue impuesta anteriormente por assume.

Evidentemente, lo que está ocurriendo es que el signo de la expresión anterior depende de -2x, es decir, de si x > 0 ó x < 0. Lo podemos comprobar:

```
--> forget(x<1)$
    assume(0<x,x<1)$
    is(expresion<0);
```

```
--> forget(0<x,x<1)$
    assume(x<0)$
    is(expresion>0);
```

```
--> forget(x<0)$
```

Para hallar máximos y mínimos relativos, calculemos los puntos críticos de f:

```
--> solve( diff(g(x),x)=0, x);
```

```
--> solve( diff(h(x),x)=0, x);
```

Esto es, h no tiene puntos críticos y g tiene un punto crítico, x = 0. Cuando x = 0 la función f es igual a g, luego éste es un punto crítico de f. Para saber si se trata de un máximo o un mínimo, podemos estudiar el signo de la derivada segunda (notemos como se calcula ésta):

```
--> diff(g(x),x,2);
```

```
--> %,x=0;
```

Por lo tanto, f''(0) = g''(0) = -2 < 0, es decir, f tiene un máximo relativo en x = 0.

Finalizaremos el estudio local, haciendo una representación gráfica de la función: Para ello, podemos definir:

```
--> /*EJEMPLO 39.4: Representación gráfica*/
    f(x):= if(x<=1) then g(x) else h(x);
```

```
--> plot2d(f(x),[x,-5,5]);
```

NOTA: La fórmula empleada para la representación gráfica de f(x) (ver figura anterior) tiene un problema: erróneamente, se representa una línea vertical en x = 1 que oculta la existencia de una discontinuidad de salto en este punto.

EJERCICIO 2:
a) Estudiar la monotonía y extremos de la curva y = x^4*exp(-x^2).

b) Estudiar la monotonía, asíntotas, extremos, concavidad, puntos de inflexión y representación gráfica de las siguientes funciones:
y = (2*x^3-5*x^4+14*x-6)/(4*x^2); y = x*exp(1/x); y = arctan(x)

5.3 POLINOMIOS DE TAYLOR

En Maxima la orden taylor(f(x),x,x0,n) muestra el desarrollo en series de Taylor hasta el término n-ésimo de la función f(x) alrededor de x0 pero no permite tratar su resultado como una expresión en x.

```
--> f(x):=sqrt(x+1);
```

```
--> taylor(f(x),x,0,2);
```

Para calcular el polinomio de Taylor como expresión se utiliza la función subst(), a continuación se calculan los polinomios de Taylor de orden 2, 3 y 4 así como sus respectivas gráficas para observar como se aproximan dichos polinomios a f(x) alrededor de x0 = 0:

```
--> /*EJEMPLO 40: Cálculo de diferentes polinomios de Taylor de la
    función anterior*/
    p2(x):=subst(t=x,taylor(f(t),t,0,2));
    p2(x);
```

```
--> p3(x):=subst(t=x,taylor(f(t),t,0,3));
    p3(x);
```

```
--> p4(x):=subst(t=x,taylor(f(t),t,0,4));
    p4(x);
```

Antes de representar graficamente los polinomios de Taylor, se desea comprobar que efectivamente los polinomios evaluados "cerca" del valor x0 = 0 producen valores cercanos a los de la función f(x). Para ello, aproximaremos la función y los polinomios anteriores en un punto cercano a x0=0. En este caso, tomaremos x1=0.3:

```
--> /*Calculamos la función f(x) en el punto x1=0.3 y lo
    comparamos con los valores obtenidos en los 3 polinomios
    de Taylor*/
    x1:0.3$
    f(x1);
    p2(x1);
    p3(x1);
    p4(x1);
```

```
--> /*Los errores de aproximación vendrán dados por*/
    abs(f(x1)-p2(x1));
    abs(f(x1)-p3(x1));
    abs(f(x1)-p4(x1));
```

Observemos que si se pretende aproximar valores algo más alejados del valor donde se está realizando el desarrollo, los polinomios de Taylor no funcionan tan bien. Por ejemplo, hacemos lo mismo anterior pero en el punto x1=1.5:

```
--> /*Calculamos la función f(x) en el punto x1=0.3 y lo
    comparamos con los valores obtenidos en los 3 polinomios
    de Taylor*/
    x2:1.5$
    f(x2);
    p2(x2);
    p3(x2);
    p4(x2);
```

```
--> /*Los errores de aproximación vendrán dados por*/
    abs(f(x2)-p2(x2));
    abs(f(x2)-p3(x2));
    abs(f(x2)-p4(x2));
```

Finalmente representamos la función f(x) y los 3 polinomios de Taylor que hemos obtenido para aproximarla:

```
-->  /*Representamos gráficamente los 3 polinomios anteriores y la
     función f(x)*/
     plot2d([f(x),p2(x),p3(x),p4(x)], [x,-2,6],[y,-1,3]);
```

EJERCICIO 3:
a) Dada la función g(x) = ln(x + 1), calcular los polinomios de Taylor de orden 2, 3 y 4 alrededor de x0=0 y representar las correspondientes gráficas para estudiar su comportamiento.

b) Se desea aproximar el número x1 = 2^(1/2). Para ello, se considera la función f(x) =(x + 3/2)^(1/2). Encuentra el polinomio de Taylor que desarrollado en un entorno de x0 = 0 aproxima a x1 con un error absoluto menor que E = 7.10^(-5).
Nota: Se considera como valor "exacto" de 2^(1/2) el dado por 2^(1/2) = 1,414213562373095.

c) Sea la función p(x) = 1 + x + x^2 ¿Cuál es su polinomio de Taylor de orden 2 alrededor de x0 = 0? ¿y de x0 = 10?, ¿Será cierto para cualquier polinomio de grado n?

6 CÁLCULO INTEGRAL EN UNA VARIABLE.

6.1 CÁLCULO DE PRIMITIVAS

En wxMaxima es posible calcular integrales de funciones mediante la orden integrate(f(x),x) o mediante el botón [Integrar. . .] e introduciendo en la ventana de diálogo la correspondiente función.

```
-->  /*EJEMPLO 41: Cálcular la primitiva de la siguiente función*/
     f(x):=3*x^2-x/3+1;
     integrate(f(x),x);
```

De la misma forma que ocurre con la orden taylor(), la orden integrate() no genera una expresión en x que nos permita realizar cálculos. De nuevo se utiliza la orden subst():

```
-->  F(x):=subst(t=x,integrate(f(t),t));
     F(x);
```

El conjunto formado por todas las primitivas de la función f(x) viene dado por:

```
-->  F(x):=subst(t=x,integrate(f(t),t))+C;
     F(x);
```

Podemos comprobar que F(x) es primitiva de f(x) derivando respecto de x:

```
-->  diff(F(x),x);
```

6.2 INTEGRAL DEFINIDA

Sabemos que define el área del recinto formado por las rectas x = a, x = b, y = 0 y la gráfica de f(x) como la integral definida de la función en valor absoluto entre los extremos a y b.

Comenzaremos calculando el área bajo la curva f(x) = e^x*cos(x) entre a = 0 y b = pi:
En primer lugar nos aseguraremos si la función cambia de signo en el intervalo indicado, pues si así fuese se debe definir abs(f(x)) como una función a trozos en intervalos, y por lo tanto la integral en dicho intervalo se divide en suma de integrales.

```
-->  /*EJEMPLO 42: Cálculo de un área de un recinto plano*/
     kill(f,g,h)$
     f(x):=%e^x*cos(x)$
     solve(f(x)=0,x);
```

El mensaje de warning indica que para resolver la ecuación se han utilizado inversas de funciones trigomométricas que unicamente están definidas en ciertos intervalos y al ser funciones periódicas existen soluciones que no se están mostrando. En el intervalo [0,pi] la función cos(x) se anula una única vez, x = pi/2, y la función exponencial no se anula nunca.

El siguiente paso es ver qué signo tiene la función f(x) en cada intervalo [0,pi/2] y [pi/2,pi]. Para lo cual se elige un punto del intervalo [0,pi/2] y se calcula su imagen por la función f(x); el signo de este valor es el signo de la función para todo ese intervalo. Se repetirá el mismo procedimiento para el segundo intervalo:

```
-->  f(1),numer;
     f(2),numer;
```

Así vemos que f(x) es positiva en el primer intervalo, y negativa en el segundo. Por ello, el valor del área pedida lo obtenemos haciendo

```
-->  integrate(f(x),x,0,%pi/2)-integrate(f(x),x,%pi/2,%pi);
```

```
-->  %,numer;
```

NOTA: En wxMaxima se puede calcular la integral definida usando el botón Análisis ->Integrar, modificando en la ventana de diálogo los límites de integración.

```
-->  /*EJEMPLO 43: Calcular el área comprendida entre las gráficas
     de f(x)=3x^3-x^2-10x y g(x)=-x^2+2x*/
     /*Calcularemos en primer lugar los puntos de corte
     (si los hay)*/

     kill(f,g)$
     f(x):=3*x^3-x^2-10*x$
     g(x):=-x^2+2*x$
     solve(f(x)=g(x),x);
```

```
-->  /*En lugar de estudiar que gráfica está por encima en cada
     uno de los intervalos (-2,0) y (0,2) nos limitaremos a
     calcular en cada uno de estos intervalos la integral de una
     función menos la otra, y tomaremos el resultado en
     valor absoluto*/

     Area1:integrate(f(x)-g(x),x,-2,0);
     Area2:integrate(f(x)-g(x),x,0,2);
```

```
-->  /*Por tanto el area total será*/

     Areacomprendida:abs(Area1)+abs(Area2);
```

En la siguiente gráfica se puede comprobar el estudio realizado:

```
-->  plot2d([f(x),g(x)],[x,-3,3]);
```

6.3 APLICACIONES DE LA INTEGRAL DEFINIDA

Vamos a ver a continuación unos sencillos ejemplos con las aplicaciones más típicas de la integral definida.

- CÁLCULO DE LONGITUDES DE CURVAS:
Sabemos que la longitud de una curva dada por y=f(x) entre dos puntos x1=a y x2=b viene dada por la expresión
L = integrate((1+(f'(x))^2)^(1/2),x,a,b)

```
-->  /*EJEMPLO 44: Calcular la longitud de la curva e^x desde a = 0
     hasta b = 10*/

     kill(f,g)$
     f(x):=%e^x;
     g(x):=subst(t=x,diff(f(x),x))$
```

```
-->  integrate(sqrt(1+g(x)^2),x,0,10),numer;
```

- VOLUMEN DE UN SÓLIDO DE REVOLUCIÓN:
Ya conocemos las fórmulas que nos permiten calcular el volumen del sólido obtenido al girar una región plana alrededor del eje OX o del eje OY, por lo que pasaremos directamente a un ejemplo:

```
-->  /*EJEMPLO 45: Calcular el volumen que se genera al girar
     alrededor del eje OX la región determinada por
     f(x)=(4-x^2)^1/2 y g(x)=(1-x^2)^1/2 entre los puntos
     a=-1 y b=1*/

     kill(f,g)$
     f(x):=sqrt(4-x^2);
     g(x):=sqrt(1-x^2);
```

```
-->  %pi*integrate(f(x)^2-g(x)^2,x,-1,1);
```

- ÁREA DE UNA SUPERFICIE DE REVOLUCIÓN:
Dada una función f(x), el área de revolución obtenida al girar la gráfica de esta función alrededor del eje OX entre x=a y x=b viene dada por
AS(OX)=2pi*integral(f(x)*(1+(f'(x))^2)^1/2) entre a y b;
mientras que si gira en torno a OY el área viene dada por
AS(OY)=2pi*integral(x*(1+(f'(x))^2)^1/2) entre a y b.

Veamos entonces un ejemplo:

```
-->  /*EJEMPLO 46: Calcular el área de revolución del sólido que
     se genera al girar la curva y =(1-x^2)^1/2 alrededor del eje OX
     entre los puntos a=-1 y b=1*/

     kill(f,g)$
     f(x):=sqrt(1-x^2);
     g(x):=subst(t=x,diff(f(x),x))$
```

```
-->  2*%pi*integrate(f(x)*sqrt(1+g(x)^2),x,-1,1);
```

EJERCICIO 4:
a) Calcular el área comprendida entre las gráficas f(x)=e^x y g(x)=4-x^2.
b) Calcular la longitud del arco de curva y =x^2x+x+1 entre a=0 y b=5.
c) Calcular el área de la superficie de revolución que se forma al girar la gráfica de la función g(x) = x alrededor del eje OX entre los puntos a = 0 y b = 10. ¿Qué superfice se forma? ¿Coincide los resultado con la fórmula de la superficie de dicha figura?.
d) Calcular el volumen del sólido de revolución que se forma al girar la región acotada por las gráficas de las funciones h1(x)=5+(1-x^2)^1/2 y h2(x)=5-(1-x^2)^1/2 alrededor del eje OX entre los puntos a=-1 y b = 1. ¿Qué sólido se forma? ¿Coincide el resultado con la fórmula del volumen de dicho sólido?.