Práctica 4:
Funciones reales de variable real: representación gráfica, derivación e integración con *Mathematica*

Grado en Administración y Dirección de Empresas

José Manuel Cascón Barbero (casbar@usal.es)
Bernardo Ramón García-Bernalt Alonso (bgarcia@usal.es)
María Dolores García Sanz (dgarcia@usal.es)
María Aurora Manrique García (amg@usal.es)
Gustavo Santos García (santos@usal.es)

1. Introducción

Ya conocemos el modo de “escribir funciones” por los capítulos previos, y también algunos comandos relacionados con su estudio. Es, no obstante, en este capítulo cuando dedicamos tiempo a esta cuestión y, en concreto, a funciones de una variable. A pesar de que en el programa de Análisis Matemático del Grado en Administración y Dirección de Empresas no hay un tema específico dedicado al estudio de funciones de una variable, sí creemos que resulta muy útil el tiempo que podamos dedicarle para a continuación, en el capítulo siguiente, abordar las funciones de varias variables, cuyo estudio constituye el grueso de la asignatura. Es de hecho una práctica habitual en las clases. En la sección 2 insistiremos en la aplicación de los comandos que permiten la representación gráfica de funciones, pues aunque ya han sido introducidos anteriormente, no podemos excluirlos ahora al, en nuestra opinión, no tener sentido aislar el estudio de las funciones de su gráfica, permitiendo además que el capítulo sea autocontenido. También revisaremos conceptos sobre simetría y traslación de funciones, y recordaremos las funciones elementales: exponencial, logarítmica y funciones trigonométricas. En la sección 3 expondremos los comandos que utiliza *Mathematica* para la resolución de ecuaciones algebraicas, tanto de forma exacta como aproximada. La sección 4 está dedicada al cálculo diferencial y algunas aplicaciones del mismo. Por último, en la sección 5 se introducirán los comandos que permiten integrar en *Mathematica*.

Para comenzar, recordamos como definir funciones con *Mathematica:*

Nombre[Argumento1_,Argumento2_,Argumento3_,...] := Definición

El **Nombre** y **Argumentos** deben cumplir las mismas reglas que se avanzaron para el nombre variables en la práctica anterior. Los argumentos deben estar entre corchetes **[ ]**, y después de cada uno de ellos debe figurar el guión bajo **_**. Esto indica a *Mathematica* que ese argumento será utilizado en la definición.

Por ejemplo:

In[1]:=
```
f[x_] := x^2 + 3 x + 5;
```

2. Representación gráfica

Mathematica permite dibujar funciones tanto vienen expresadas en forma explícita (y = f(x)), como en forma implícita (F(x,y) = 0) o paramétrica (x = x(t), y = y(t))

2.1 Funciones en forma explícita (y =f(x))

Para representar funciones de una variable real se utiliza el comando **Plot**:

In[2]:=
```
? Plot
```

Out[2]=

Symbol

Plot[f, {x, x_{min}, x_{max}}] generates a plot of f as a function of x from x_{min} to x_{max}.

Plot[{f_1, f_2, …}, {x, x_{min}, x_{max}}] plots several functions f_i.

Plot[{…, w[f_i], …}, …] plots f_i with features defined by the symbolic wrapper w.

Plot[…, {x} ∈ reg] takes the variable x to be in the geometric region reg.

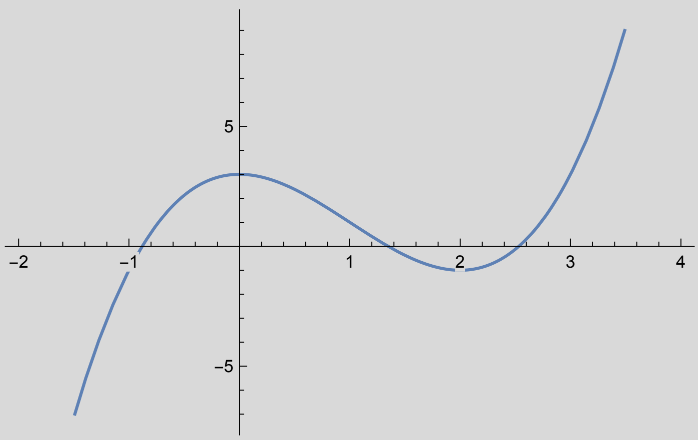
Aquí **f** es la función que queremos dibujar, **x** es la variable, mientras x_{min}, x_{max}, denotan los extremos del intervalo donde queremos dibujar la función.

Ejemplo. Dibujar la función $f(x) = x^3 - 3x^2 + 3$ en el intervalo [-2,4]

In[3]:=
```
f[x_] := x^3 - 3 x^2 + 3;
```

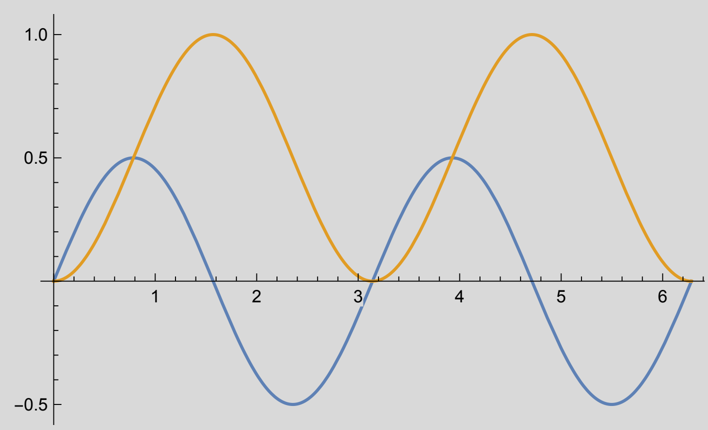
In[4]:=
```
Plot[f[x], {x, -2, 4}]
```
representación gráfica

Out[4]=



El comando permite dibujar varias funciones en el mismo gráfico, para ello se pasan en forma de lista (es decir, entre llaves { }).

In[5]:=
```
Plot[{Cos[x] Sin[x], Sin[x]^2}, {x, 0, 2 Pi}]
```

Out[5]=


Existen multitud de opciones para dar forma a los gráficos; se puede acceder a ellas a través del *menú rápido* que aparece tras realizar un gráfico, accediendo a la ayuda o consultando ejemplos. No mostramos aquí ninguna opción de este tipo, sino que irán apareciendo en el documento a medida que las necesitemos.

2.2 Funciones en forma implícita (F(x,y) = 0)

En el caso de funciones en forma implícita el comando que debemos usar es **ContourPlot**:

In[6]:=
```
? ContourPlot
```

Out[6]=

Symbol

ContourPlot[f, {x, x_{min}, x_{max}}, {y, y_{min}, y_{max}}] generates a contour plot of f as a function of x and y.

ContourPlot[$f == g$, {x, x_{min}, x_{max}}, {y, y_{min}, y_{max}}] plots contour lines for which $f = g$.

ContourPlot[{$f_1 == g_1$, $f_2 == g_2$, …}, {x, x_{min}, x_{max}}, {y, y_{min}, y_{max}}] plots several contour lines.

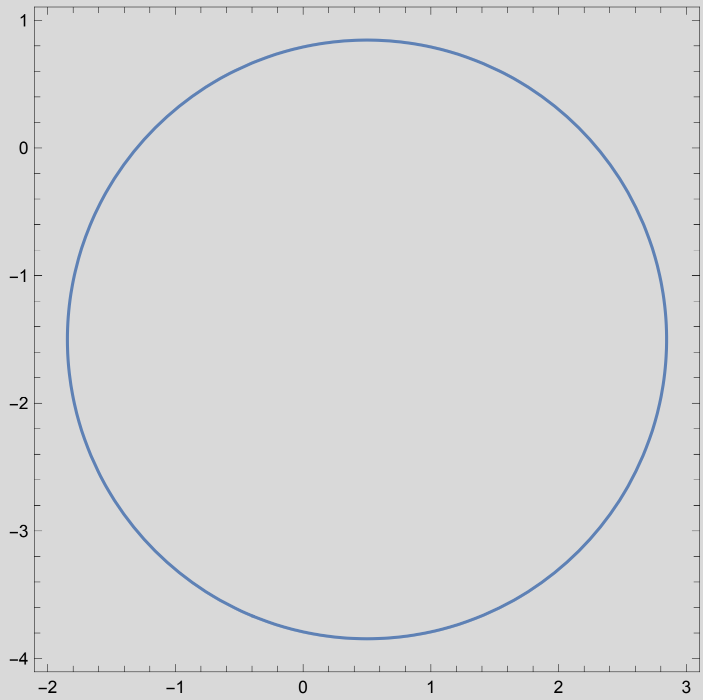
ContourPlot[…, {x, y} ∈ reg] takes the variables {x, y} to be in the geometric region reg.

En nuestro caso, debemos colocar en el primer argumento la función **F(x,y) ==0** (recuerda que el doble igual **==** se utiliza en *Mathematica* para comparar y el **=** simple para asignar), y en el segundo y tercer argumento, las variables **(x,y)** y su rango de variación, respectivamente.

Ejemplo. Dibujar la circunferencia $x^2 + y^2 - x + 3y + 3 = 0$

In[7]:=
```
ContourPlot[x^2 + y^2 - x + 3 y - 3 == 0 , {x, -2, 3}, {y, -4, 1}]
representación de contornos
```

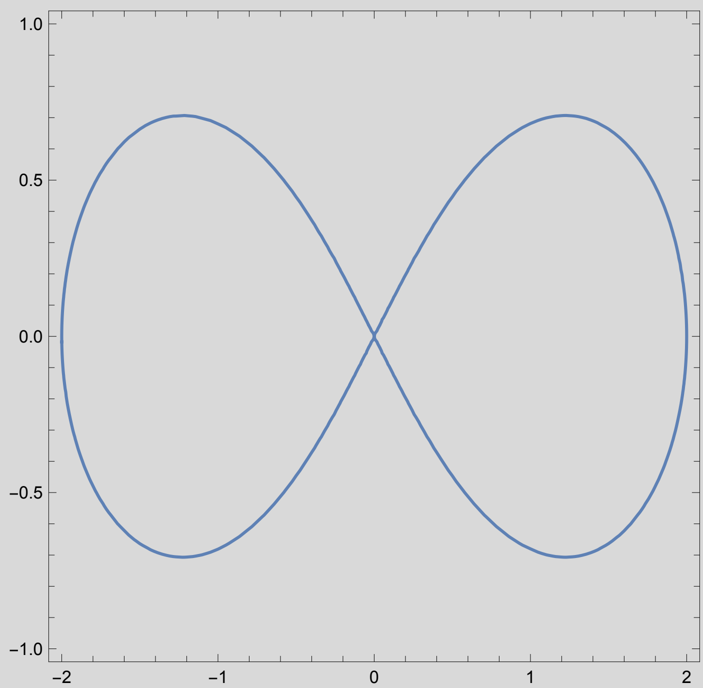
Out[7]=



En general *Mathematica* dibujará cualquier función de este tipo, por muy complicada que nos parezca la fórmula.

In[8]:=
```
ContourPlot[(x^2 + y^2)^2 == 4 (x^2 - y^2), {x, -2, 2}, {y, -1, 1}]
representación de contornos
```

Out[8]=



2.3 Funciones en forma paramétrica (x= x(t), y= y(t))

En el caso de las funciones en forma paramétrica, las coordenadas (x,y) son funciones de un parámetro (piensa, por ejemplo, en una partícula que se mueve en función del tiempo). *Mathematica* dispone en este caso de la función **ParametricPlot**

In[9]:=
```
? ParametricPlot
```

Out[9]=

Symbol

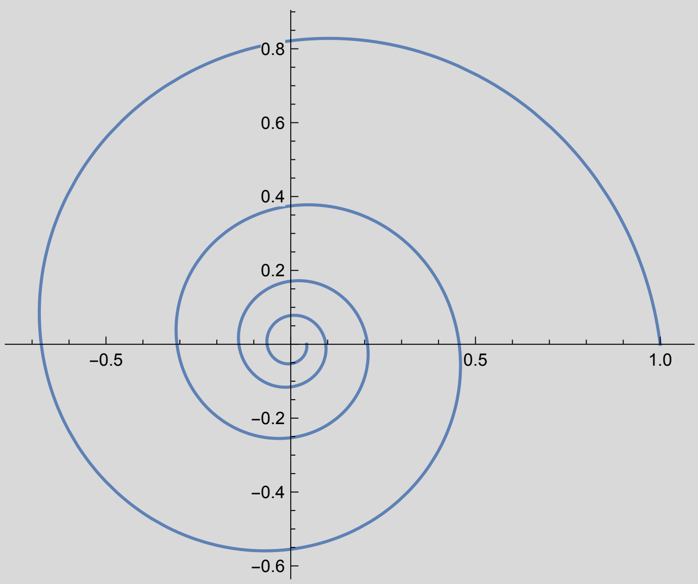
ParametricPlot[{f_x, f_y}, {u, u_{min}, u_{max}}] generates a parametric plot of a curve with x and y coordinates f_x and f_y as a function of u.
ParametricPlot[{{f_x, f_y}, {g_x, g_y}, …}, {u, u_{min}, u_{max}}] plots several parametric curves.
ParametricPlot[{f_x, f_y}, {u, u_{min}, u_{max}}, {v, v_{min}, v_{max}}] plots a parametric region.
ParametricPlot[{{f_x, f_y}, {g_x, g_y}, …}, {u, u_{min}, u_{max}}, {v, v_{min}, v_{max}}] plots several parametric regions.
ParametricPlot[{…, w[{f_x, f_y}], …}, …] plots the curve {f_x, f_y} with features defined by the symbolic wrapper w.
ParametricPlot[…, {u, v} ∈ reg] takes parameters {u, v} to be in the geometric region reg.

El primer argumento son las funciones coordenadas en función del parámetro. El segundo argumento corresponde al parámetro y al rango de estudio

Ejemplo. Dibujar la espiral dada por las ecuaciones: (x(t) $= e^{-t/8} \cos(x)$, $y(t) = e^{-t/8} \sin(x)$ en el intervalo t $\in [0, 8\pi]$

In[10]:=
```
ParametricPlot[{Exp[-t/8] Cos[t], Exp[-t/8] Sin[t]}, {t, 0, 8 Pi}]
```
gráfico paramétrico | exponencial | coseno | exponencial | seno | númer

Out[10]=



2.4 Funciones elementales: Exponencial, Logaritmo, Seno, Coseno

En el material teórico de la asignatura se han revisado este tipo de funciones. Utilizaremos esta sección para ver su representación gráfica mediante *Mathematica:*

2.4.1 Exponencial

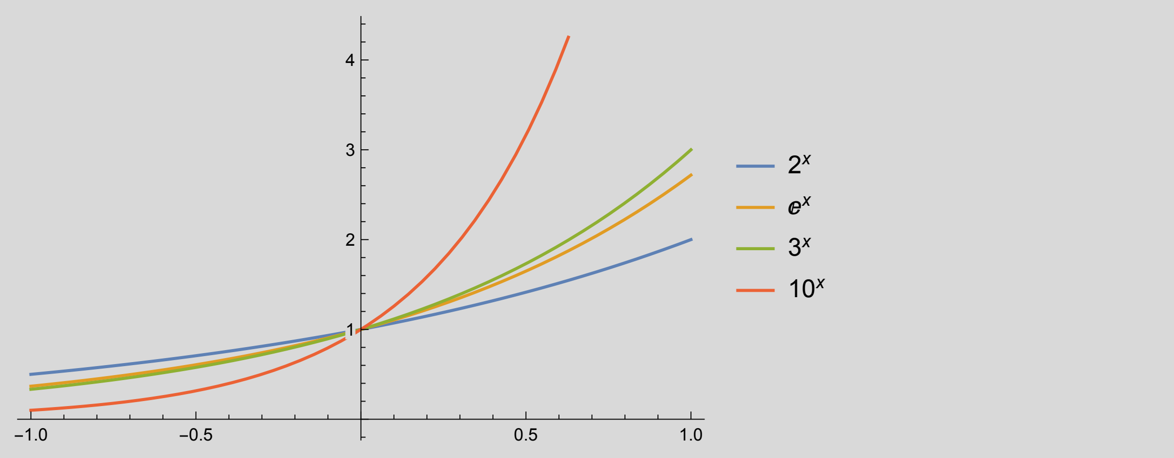
Dibujamos a continuación varias funciones exponenciales considerando diferentes bases. Observa las características de cada una de ellas

In[11]:=

```
Plot[{2^x, E^x, 3^x, 10^x}, {x, -1, 1}, PlotLegends → "Expressions"]
```

representación·· número e leyendas de representación

Out[11]=

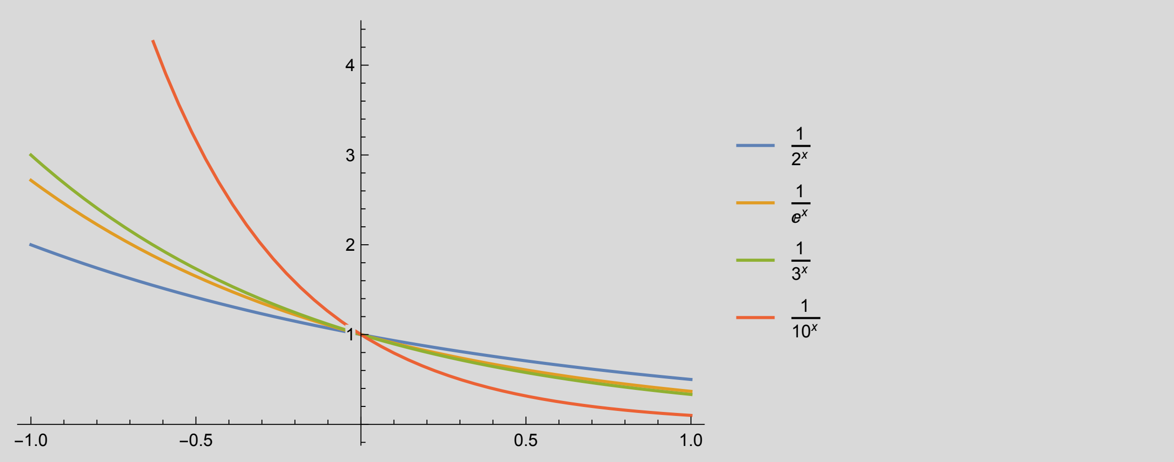


In[12]:=

```
Plot[{1/2^x, 1/E^x, 1/3^x, 1/10^x},
  {x, -1, 1}, PlotLegends → "Expressions"]
```

representación gráfica número e leyendas de representación

Out[12]=



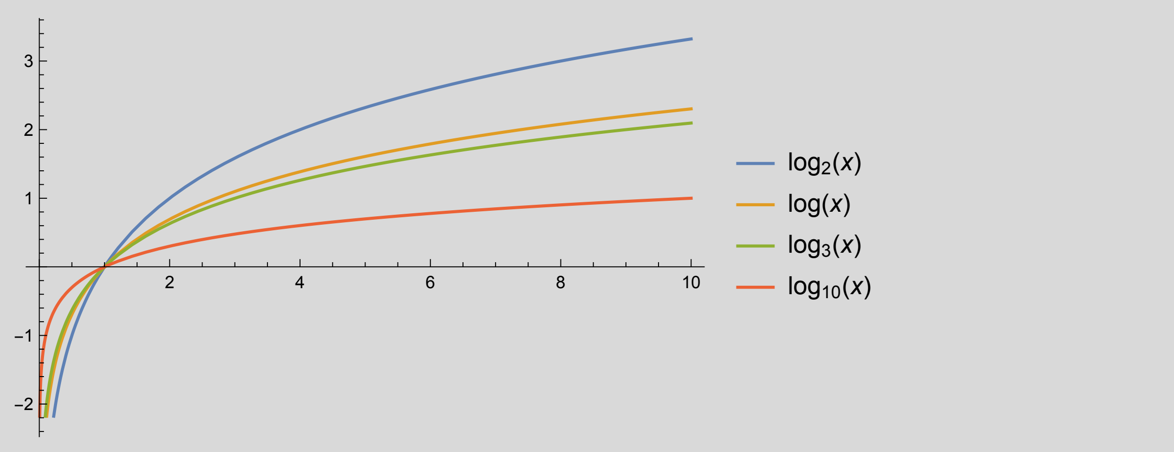
2.4.2 Logaritmo

En *Mathematica* **Log[x]** denota el logaritmo neperiano o natural (en base **E**). Si deseamos utilizar otro logaritmo la forma de hacerlo será mediante el comando **Log[b,x]**, donde **b** denota la base (es decir **Log[x] = Log[E,x]**). Realizamos ahora el mismo ejercicio que en la sección anterior para el caso de la función logaritmo.

In[13]:=

```
Plot[{Log[2, x], Log[x], Log[3, x], Log[ 10, x]},
⌊repre··· ⌊logaritmo   ⌊logaritmo ⌊logaritmo   ⌊logaritmo
 {x, 0, 10}, PlotLegends → "Expressions"]
            ⌊leyendas de representación
```

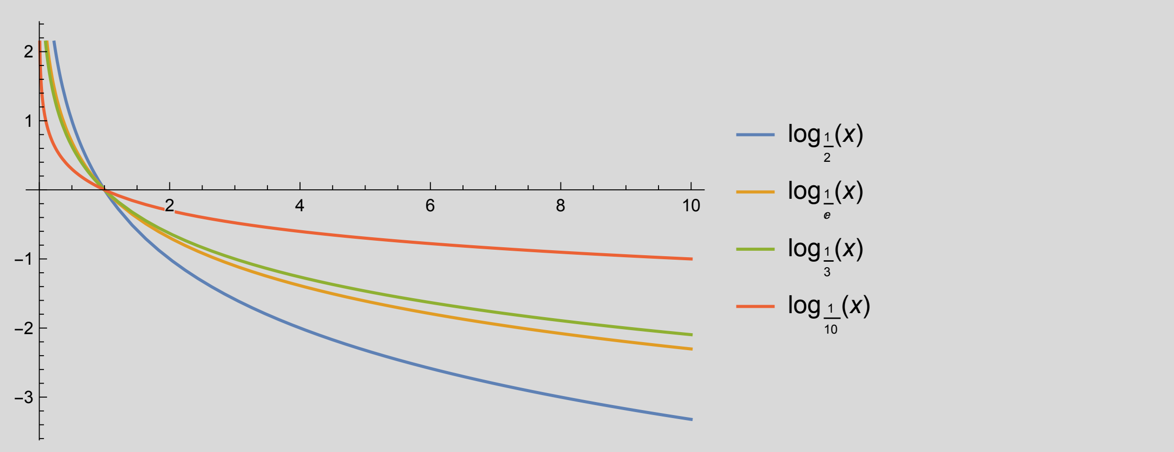
Out[13]=



In[14]:=

```
Plot[{Log[1/2, x], Log[1/E, x], Log[1/3, x], Log[ 1/10, x]},
⌊repre··· ⌊logaritmo      ⌊logaritmo ⌊númer··· ⌊logaritmo      ⌊logaritmo
 {x, 0, 10}, PlotLegends → "Expressions"]
            ⌊leyendas de representación
```

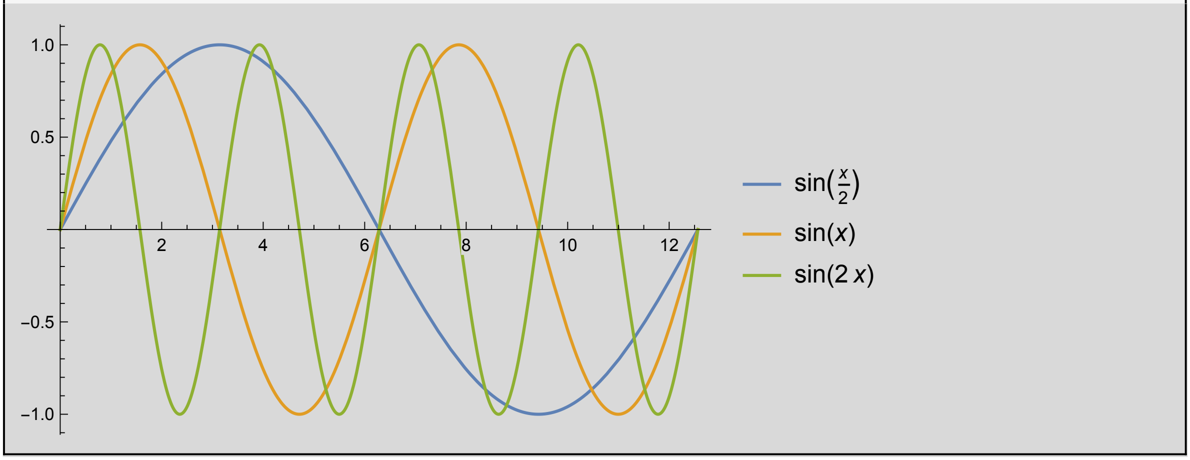
Out[14]=



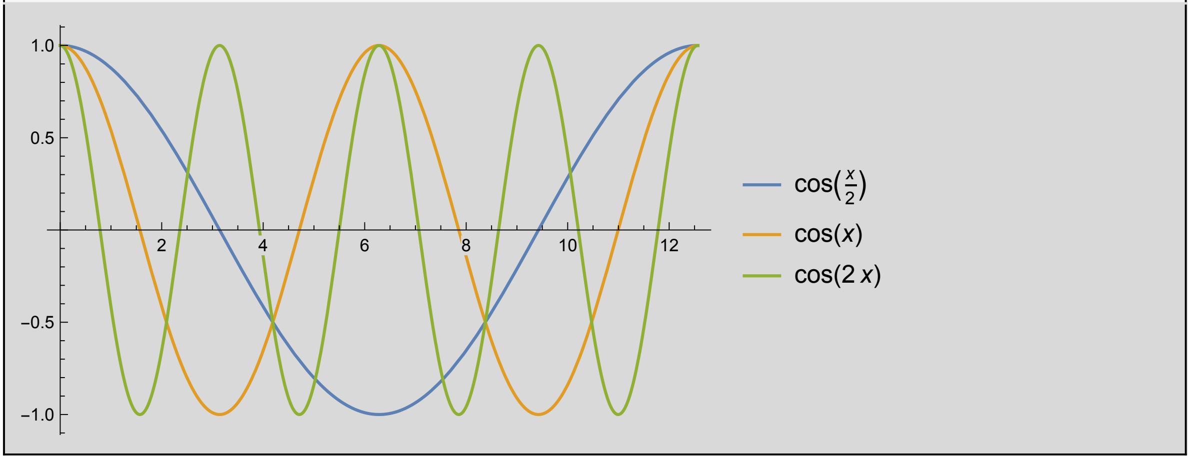
2.4.3 Trigonométricas: Seno, Coseno

En *Mathematica* **Sin[x]** denota la función seno y **Cos[x]**, la función coseno. Dibujemos varios ejemplos:

In[15]:=
```
Plot[{Sin[x/2], Sin[x], Sin[2 x]}, {x, 0, 4 Pi}, PlotLegends → "Expressions"]
```
⌊repre··· ⌊seno ⌊seno ⌊seno ⌊nú··· ⌊leyendas de representación

Out[15]=


In[16]:=
```
Plot[{Cos[x/2], Cos[x], Cos[2 x]}, {x, 0, 4 Pi}, PlotLegends → "Expressions"]
```
⌊repre··· ⌊coseno ⌊coseno ⌊coseno ⌊nú··· ⌊leyendas de representación

Out[16]=


2.5 Simetrías y traslaciones

Debemos recordar que dada una función f(x): R →R, se denomina:

- Función simétrica de f(x) respecto al eje OX a la función -f(x)
- Función simétrica de f(x) respecto al eje OY a la función f(-x)
- Función simétrica de f(x) respecto al origen a la función -f(-x)

Además se dice que una función es par si es simétrica respecto al eje OY, es decir si f(x) = f(-x), e impar si es simétrica respecto al origen, es decir si f(x) =-f(-x)

Ejemplos.

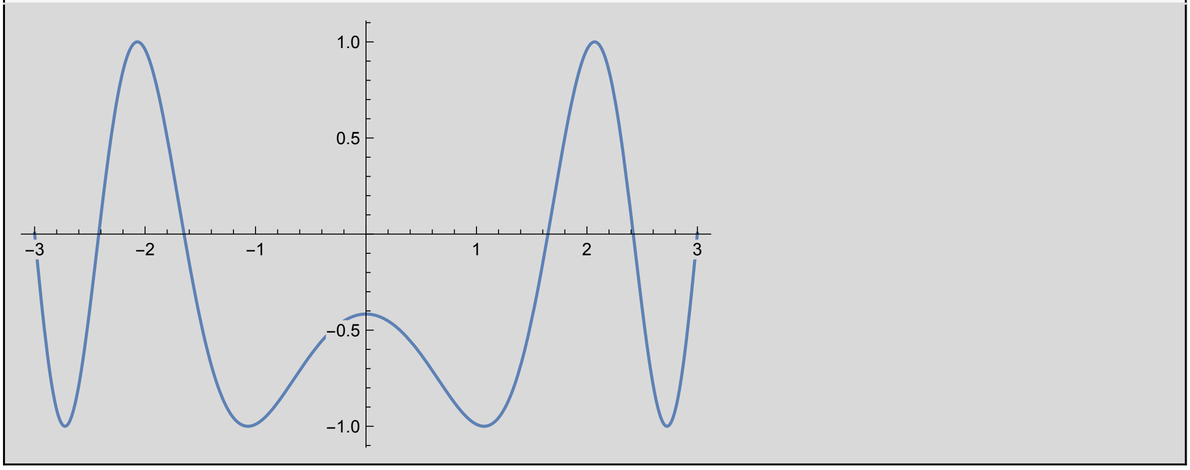
- La función f(x) = cos(x^2 + 2) es par:

In[17]:=

```
Plot[Cos[x^2 + 2], {x, -3, 3}]
```

repr··· coseno

Out[17]=



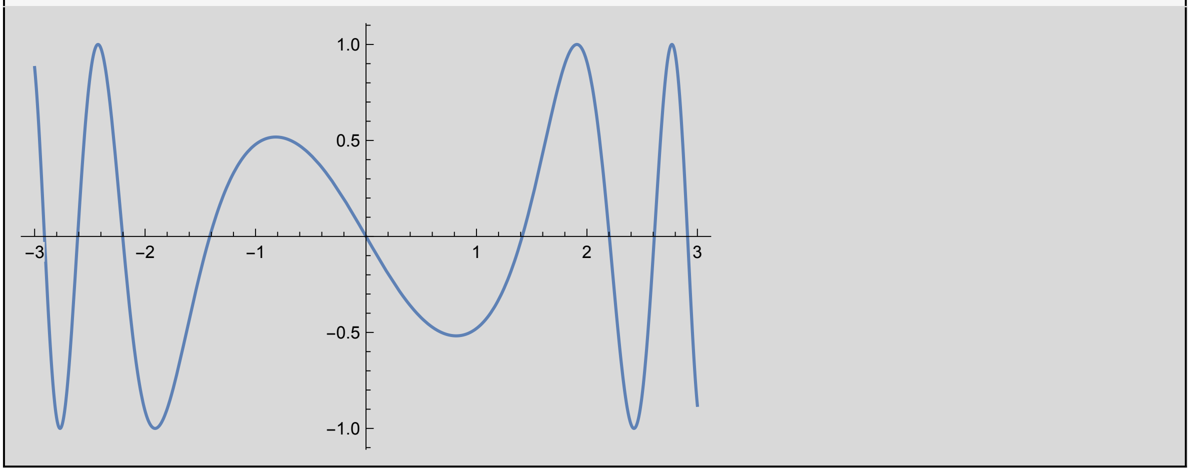
- La función f(x) = sin($\frac{x^3}{2}$-x) es impar:

In[18]:=

```
Plot[Sin[x^3/2 - x], {x, -3, 3}]
```

repr··· seno

Out[18]=



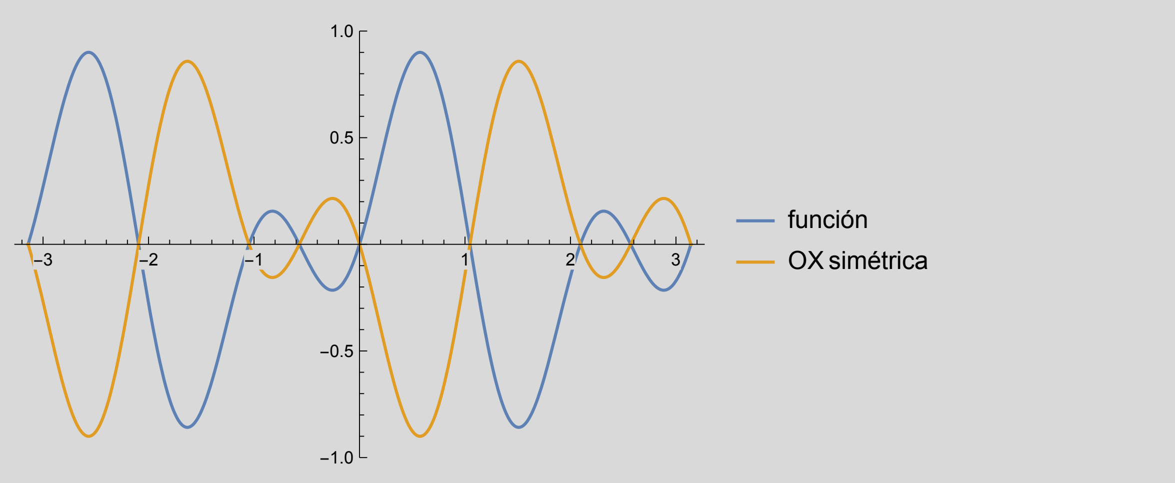
Ejercicio. Dada la función y = sin(3x) cos(x-1), utiliza *Mathematica* para calcular las funciones simétricas de f(x) respecto al eje OX, al eje OY y al origen, en el intervalo [-π,π]:

In[19]:=

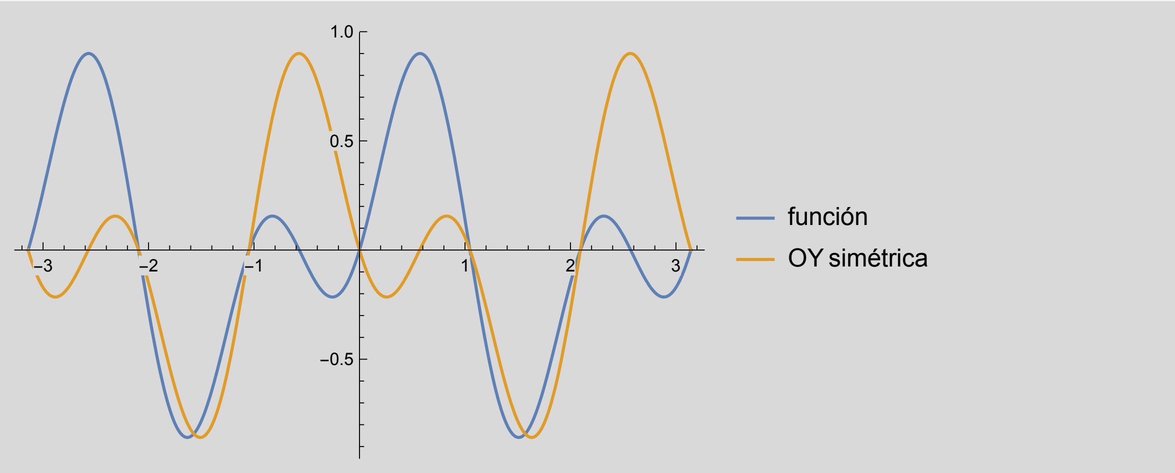
```
f[x_] := Sin[3 x] Cos[x - 1]
```

seno coseno

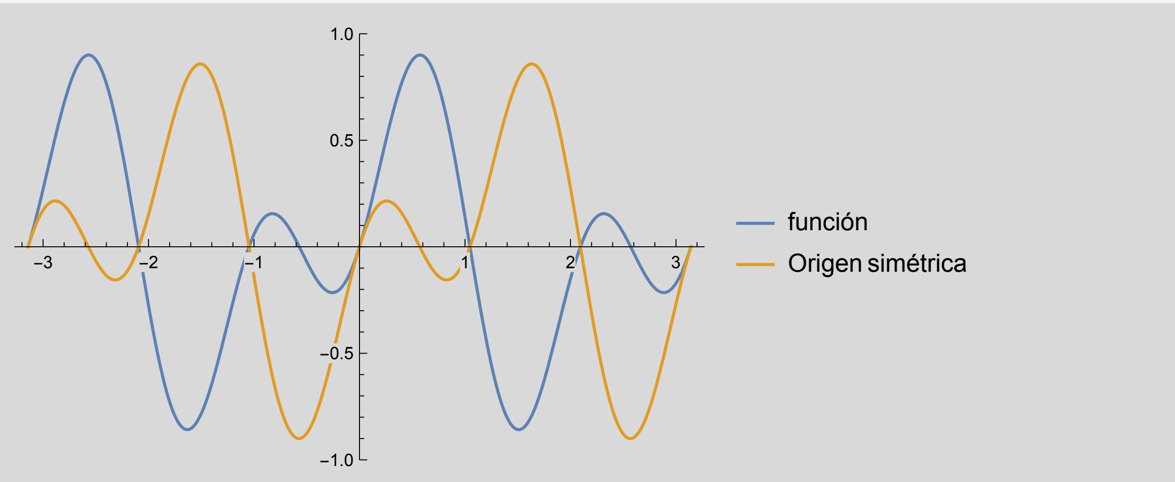
In[20]:=
```
Plot[{f[x], -f[x]}, {x, -Pi, Pi}, PlotLegends → {función, simétrica OX}]
```
representación gráfica n··· nú··· leyendas de representación

Out[20]=


In[21]:=
```
Plot[{f[x], f[-x]}, {x, -Pi, Pi}, PlotLegends → {función, simétrica OY }]
```
representación gráfica n··· nú··· leyendas de representación

Out[21]=


In[22]:=
```
Plot[{f[x], -f[-x]}, {x, -Pi, Pi}, PlotLegends → {función, simétrica Origen }]
```
representación gráfica n··· nú··· leyendas de representación

Out[22]=


Con respecto a las traslaciones, recuerda que:

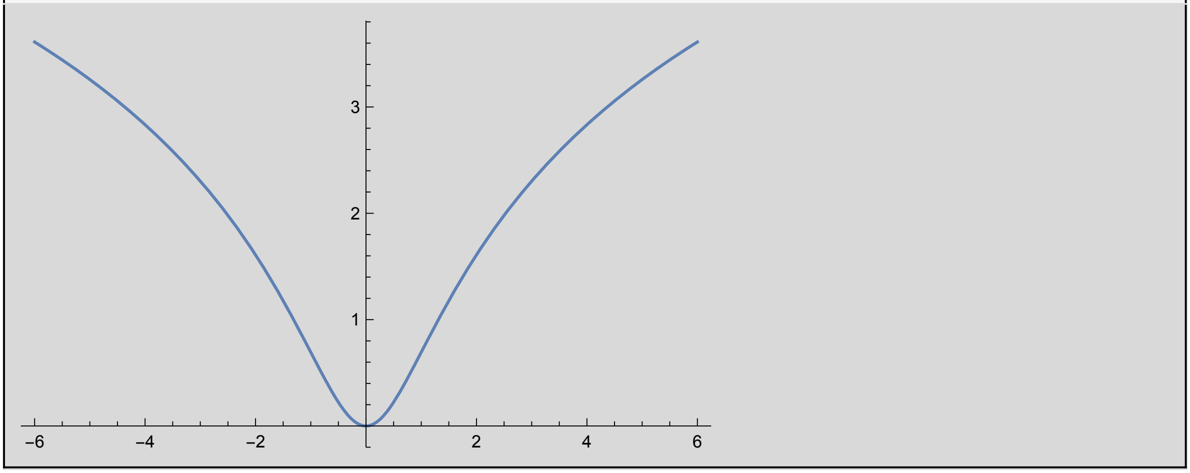
- f(x-a) es una traslación horizontal de f(x) en a unidades
- a+ f(x) es una traslación vertical de f(x) en a unidades

Ejemplo. Dada la función f(x)= $\log(x^2 + 1)$

- a) Calcular traslaciones horizontales de f(x) de 1 y 2 unidades
- b) Calcular traslaciones verticales de f(x) de 1 y 2 unidades
- c) Calcular la traslación de f(x) de 2 unidades en dirección vertical y -3 unidades en dirección horizontal

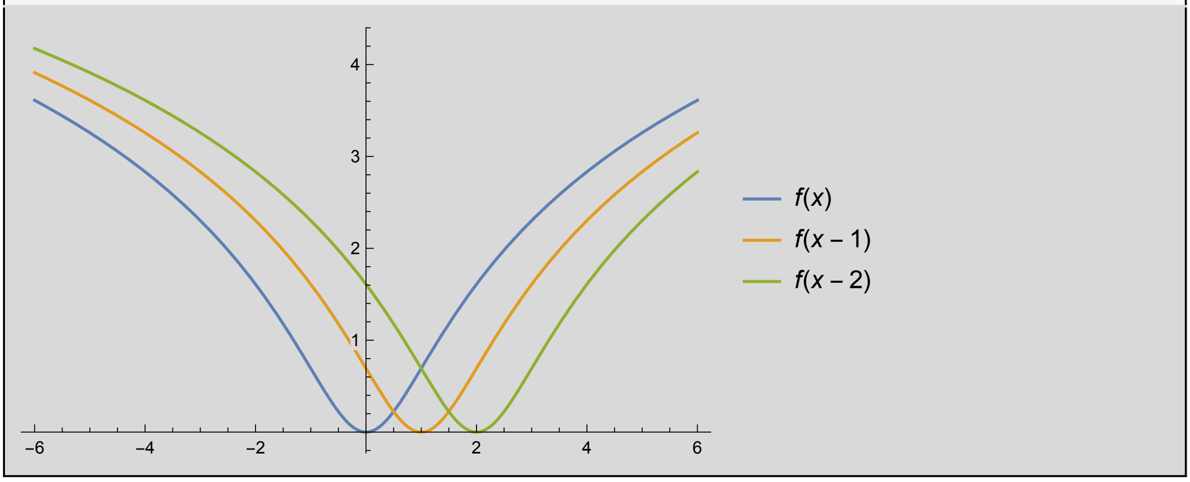
In[23]:=
```
f[x_] := Log[x^2 + 1]
```
logaritmo

In[24]:=
```
Plot[f[x], {x, -6, 6}]
```
representación gráfica

Out[24]=


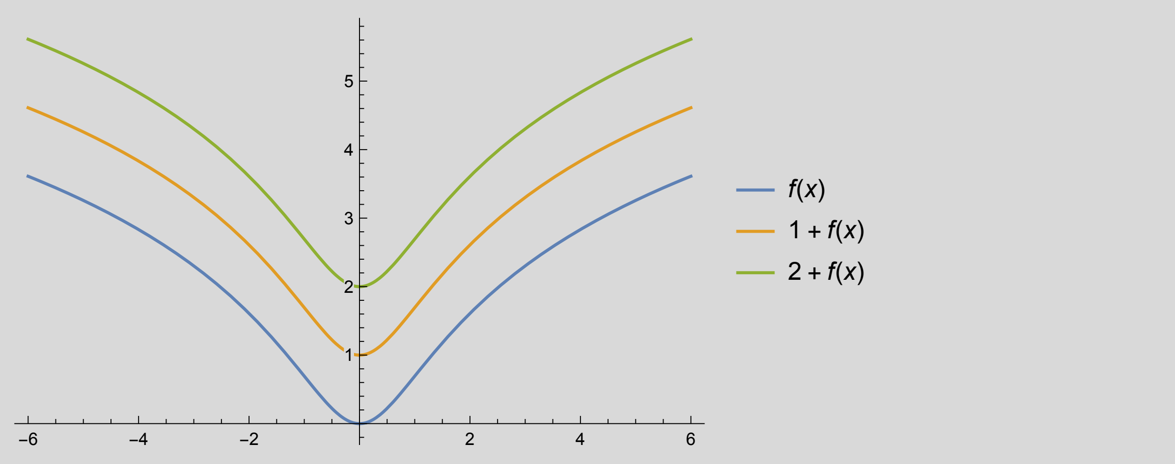
Caso a)

In[25]:=
```
Plot[{f[x], f[x - 1], f[x - 2]}, {x, -6, 6}, PlotLegends -> "Expressions"]
```
representación gráfica
leyendas de representación

Out[25]=


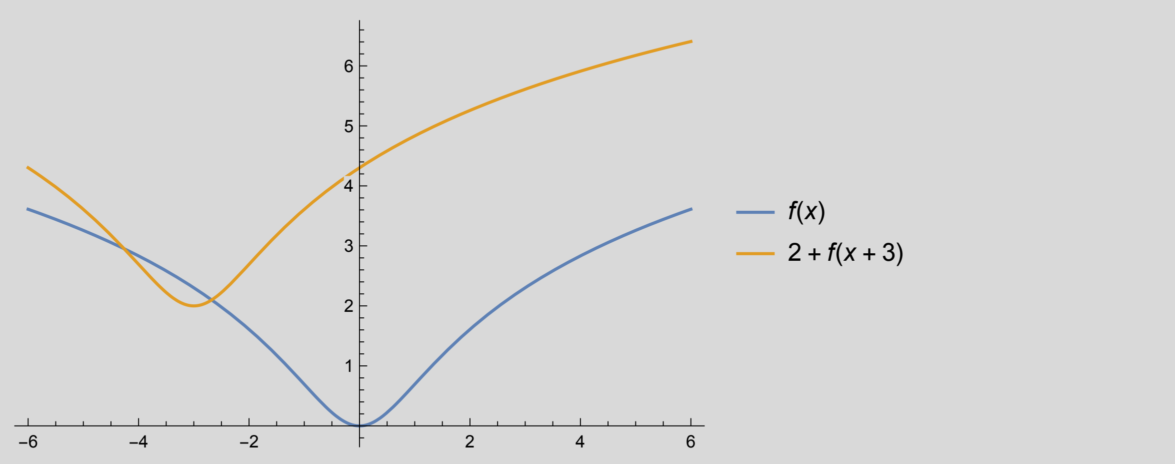
Caso b)

In[26]:=
```
Plot[{f[x], 1 + f[x], 2 + f[x]}, {x, -6, 6}, PlotLegends -> "Expressions"]
```
representación gráfica leyendas de representación

Out[26]=


Caso c)

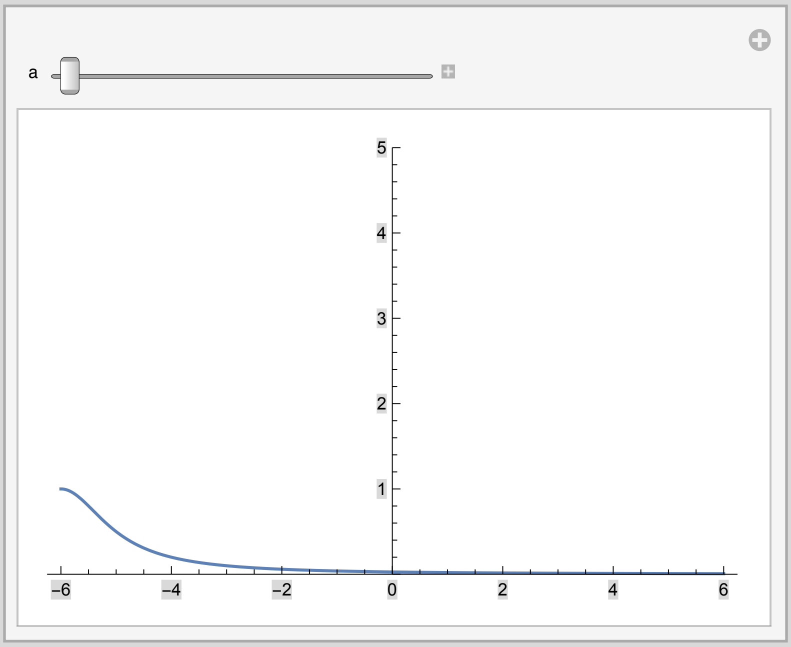
In[27]:=
```
Plot[{f[x], 2 + f[x + 3]}, {x, -6, 6}, PlotLegends -> "Expressions"]
```
representación gráfica leyendas de representación

Out[27]=


Para finalizar esta sección, podemos usar la función **Manipulate**, que genera animaciones en función de un parámetro, para observar el efecto de las traslaciones horizontales y verticales:

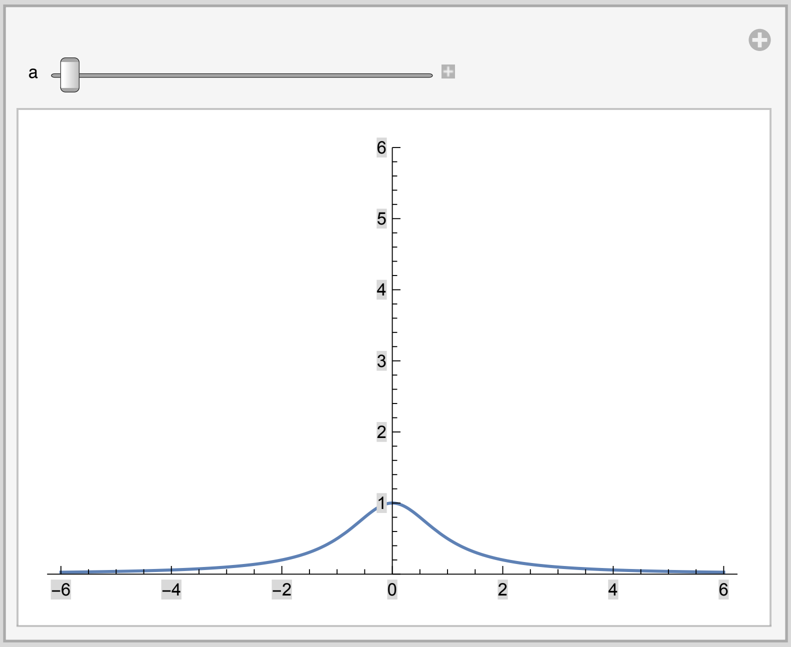
In[28]:=
```
Manipulate[Plot[f[x - a], {x, -6, 6}, PlotRange → {0, 5}], {a, -6, 6}]
```
manipula · representación gráfica · rango de representación

Out[28]=



In[29]:=
```
Manipulate[
 Plot[a + f[x], {x, -6, 6}, AxesOrigin → {0, 0}, PlotRange → {0, 6}], {a, 0, 4}]
```
manipula · representación gráfica · origen de ejes · rango de representación

Out[29]=



3. Resolución de ecuaciones

Mathematica proporciona varias funciones para la resolución de ecuaciones, desde aquellas de propósito específico -por ejemplo **LinearSolve** está creada para resolver sistemas lineales de ecuaciones , y **Roots** para calcular raíces de polinomios- a otras basadas en métodos numéricos como el de Newton (por ejemplo **FindRoot**). En esta práctica nos centraremos solo en las más genéricas: **Solve**, **Reduce** y su variante numérica **NSolve**.

3.1 Resolución analítica

3.1.1 Función **Solve**

La sintaxis de la función **Solve** es la siguiente:

In[30]:=
```
? Solve
```

Out[30]=

Symbol

Solve[*expr*, *vars*] attempts to solve the system *expr* of equations or inequalities for the variables *vars*.
Solve[*expr*, *vars*, *dom*] solves over the domain
dom. Common choices of *dom* are Reals, Integers, and Complexes.

Aquí *expr*, denota la ecuación (o ecuaciones agrupadas entre llaves), y *vars* las variables involucradas. En esta práctica trabajaremos solo con ecuaciones de una variable.

Mostramos a continuación varios ejemplos:

In[31]:=
```
Solve[x^3 - 3 x + 2 == 0, x]
resuelve
```

Out[31]=
```
{{x → -2}, {x → 1}, {x → 1}}
```

La solución se proporciona como una lista de reglas de trasformación. Mas adelante veremos cómo usarlo.

La solución puede ser realmente compleja:

In[32]:=
```
Solve[x^4 + x^2 - 5 x - 3 == 0, x]
resuelve
```

Out[32]=
```
{{x → -0.529...}, {x → 1.71...},
 {x → -0.593... - 1.72... i}, {x → -0.593... + 1.72... i}}
```

La función **Solve** trata de despejar la variable *x*, para ello utiliza utiliza sobre todo funciones inversas. En ocasiones **Solve** devuelve mensajes del tipo:

In[33]:=
```
Solve[x + Exp[x] == 0, x]
resuelve   exponencial
```

Solve: Inverse functions are being used by Solve, so some solutions may not be found; use Reduce for complete solution information.

Out[33]=
```
{{x → -ProductLog[1]}}
```

que avisa al usuario que alguna solución podría haberse *perdido* durante el cálculo. Recuerda que para aproximar el valor de la solución podemos emplear **//N**

In[34]:=
```
Solve[x + Exp[x] == 0, x] // N
resuelve   exponencial   valor numérico
```

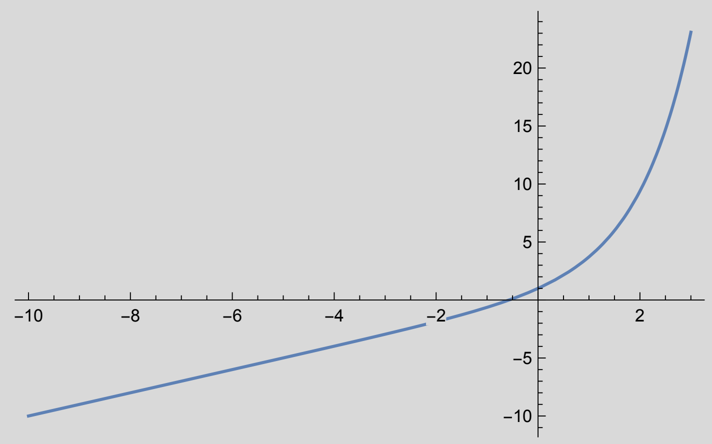
Solve: Inverse functions are being used by Solve, so some solutions may not be found; use Reduce for complete solution information.

Out[34]=
```
{{x → -0.567143}}
```

En este caso, tras dibujar la función podríamos aventurar que la ecuación solo tiene una raíz (de hecho esto se puede demostrar rigurosamente, simplemente con observar que la función es monótona creciente).

In[35]:=
```
Plot[x + Exp[x], {x, 3, -10}]
representa·· exponencial
```

Out[35]=



Sin embargo, en el caso siguiente *Mathematica* ha perdido soluciones:

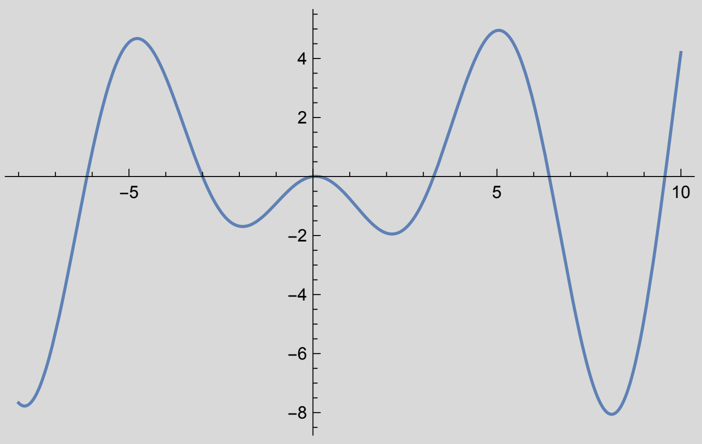
In[36]:=
```
Solve[x Sin[x + 3] == 0, x]
resuelve   seno
```

Solve: Inverse functions are being used by Solve, so some solutions may not be found; use Reduce for complete solution information.

Out[36]=
```
{{x → -3}, {x → 0}}
```

In[37]:=
```
Plot[x Sin[x + 3], {x, -8, 10}]
```
repres··· seno

Out[37]=



Para calcular las otras soluciones deberíamos utilizar otro tipo de métodos que fijan un intervalo de búsqueda, o parten de una semilla próxima a la raíz. Los introduciremos cuando sea necesario.

3.1.2 Función **Reduce**

El comando **Reduce** tiene sintaxis similar a **Solve**:

In[38]:=
```
? Reduce
```

Out[38]=

Symbol

Reduce[*expr*, *vars*] reduces the statement *expr* by solving equations or inequalities for *vars* and eliminating quantifiers.
Reduce[*expr*, *vars*, *dom*] does the reduction over the domain *dom*. Common choices of *dom* are Reals, Integers, and Complexes.

Este comando reduce la ecuación a otra similar más sencilla, mediante el uso de complejas reglas de transformación. Al igual que **Solve**, es capaz de trabajar con parámetros.
Veamos la diferencia entre ambos comandos al resolver la ecuación general de segundo grado:

In[39]:=
```
Reduce[a x^2 + b x + c == 0 , x]
```
reduce

Out[39]=

$$\left(a \neq 0 \,\&\&\, \left(x == \frac{-b-\sqrt{b^2-4ac}}{2a} \,||\, x == \frac{-b+\sqrt{b^2-4ac}}{2a}\right)\right) \,||\, \left(a == 0 \,\&\&\, b \neq 0 \,\&\&\, x == -\frac{c}{b}\right) \,||\, \left(c == 0 \,\&\&\, b == 0 \,\&\&\, a == 0\right)$$

In[40]:=
```
Solve[a x^2 + b x + c == 0, x]
```
resuelve

Out[40]=

$$\left\{\left\{x \to \frac{-b-\sqrt{b^2-4ac}}{2a}\right\}, \left\{x \to \frac{-b+\sqrt{b^2-4ac}}{2a}\right\}\right\}$$

Observa que **Solve** no analiza el valor de los parámetros.

3.2 Resolución numérica

En ocasiones las soluciones de una ecuación no pueden expresarse por medio de operadores elementales (sumas, restas, productos, cociente, radicales), y por tanto la función **Solve** poco puede hacer (en realidad, en este caso está resolviendo la ecuación a través del comando **Roots**):

In[41]:=
```
Solve[x^5 + 3 x^4 + x^2 - 5 x - 3 == 0, x]
resuelve
```
Out[41]=
```
{{x → -3.22...}, {x → -0.513...}, {x → 1.16...},
 {x → -0.213... - 1.24... i}, {x → -0.213... + 1.24... i}}
```

Para ello, podemos emplear el comando **NSolve**, con la misma sintaxis que **Solve**.

In[42]:=
```
NSolve[x^5 + 3 x^4 + x^2 - 5 x - 3 == 0, x]
resolvedor numérico
```
Out[42]=
```
{{x → -3.21854}, {x → -0.512943}, {x → -0.212698 - 1.23511 i},
 {x → -0.212698 + 1.23511 i}, {x → 1.15688}}
```

En ocasiones solo estamos interesados en calcular/aproximar las raíces reales de la ecuación. Para ello basta con añadir el parámetro *Reals*, al final de la instrucción:

In[43]:=
```
NSolve[x^5 + 3 x^4 + x^2 - 5 x - 3 == 0, x, Reals]
resolvedor numérico                      números r
```
Out[43]=
```
{{x → -3.21854}, {x → -0.512943}, {x → 1.15688}}
```

De hecho, es recomendable utilizarlo si no nos interesan las soluciones complejas, pues la resolución será más rápida.

Nota: En este caso, también habríamos podido emplear el comando:

In[44]:=
```
Solve[x^5 + 3 x^4 + x^2 - 5 x - 3 == 0, x] // N
resuelve                                  valor numérico
```
Out[44]=
```
{{x → -3.21854}, {x → -0.512943}, {x → 1.15688},
 {x → -0.212698 - 1.23511 i}, {x → -0.212698 + 1.23511 i}}
```

Sin embargo, esta alternativa no será siempre viable. Pues **Solve//N** primero resuelve con **Solve** y después aproxima, sin embargo **NSolve** utiliza procedimientos numéricos desde el principio. Además **NSolve** será mas rápido que **Solve** cuando surgen dificultades.

4. Cálculo diferencial

4.1 Cálculo de derivadas

Para el cálculo de derivadas *Mathematica* dispone del comando **D**

In[45]:=
```
? D
```

Out[45]=

Symbol

D[f, x] gives the partial derivative $\partial f/\partial x$.

D[f, {x, n}] gives the multiple derivative $\partial^n f/\partial x^n$.

D[f, x, y, …] gives the partial derivative $\cdots(\partial/\partial y)(\partial/\partial x) f$.

D[f, {x, n}, {y, m}, …] gives the multiple partial derivative $\cdots(\partial^m/\partial y^m)(\partial^n/\partial x^n) f$.

D[f, {{x_1, x_2, …}}] for a scalar f gives the vector derivative $(\partial f/\partial x_1, \partial f/\partial x_2, \ldots)$.

D[f, {$array$}] gives an array derivative.

Por ejemplo, dada la función $y = \frac{x}{x^2+1}$, la primera derivada se calcula simplemente:

In[46]:=
```
D[x / (x^2 + 1), x]
deriva
```

Out[46]=

$$-\frac{2\,x^2}{\left(1+x^2\right)^2}+\frac{1}{1+x^2}$$

En ocasiones, puede ser muy útil anteponer el comando **Simplify** para, obviamente, simplificar la expresión:

In[47]:=
```
Simplify[D[x / (x^2 + 1), x]]
simplifica deriva
```

Out[47]=

$$\frac{1-x^2}{\left(1+x^2\right)^2}$$

Para derivadas de orden superior, en el segundo argumento de D, se coloca entre llaves la variable y el orden de derivación. Así la derivada segunda se obtiene:

In[48]:=
```
Simplify[D[x / (x^2 + 1), {x, 2}]]
simplifica deriva
```

Out[48]=

$$\frac{2\,x\left(-3+x^2\right)}{\left(1+x^2\right)^3}$$

la derivada quinta:

In[49]:=
```
Simplify[D[x / (x^2 + 1), {x, 5}]]
```
Out[49]=

$$-\frac{120\left(-1+15x^2-15x^4+x^6\right)}{\left(1+x^2\right)^6}$$

o la decimoctava:

In[50]:=
```
Simplify[D[x / (x^2 + 1), {x, 18}]]
```
Out[50]=

$$\frac{1}{\left(1+x^2\right)^{19}}6\,402\,373\,705\,728\,000\,x\left(-19+969x^2-11\,628x^4+50\,388x^6-92\,378x^8+75\,582x^{10}-27\,132x^{12}+3876x^{14}-171x^{16}+x^{18}\right)$$

4.2 Cálculo de máximos, mínimos y puntos de inflexión

Dado que todos los aspectos teóricos ya fueron tratados en clase, y que ya hemos introducido los elementos necesarios para resolver este problema (**D**, **Solve**, **NSolve**), esta sección se limitará a un ejemplo concreto.

Ejemplo. Determinar los máximos, mínimos y puntos de inflexión de la función: $f(x) = x^5 + x^4 - 11x^3 - 9x^2 + 18x + 10$

Definimos, en primer lugar, la función y sus derivadas:

In[51]:=
```
f[x_] := x^5 + x^4 - 11 x^3 - 9 x^2 + 18 x + 10;
```
In[52]:=
```
df[x_] = D[f[x], x]
```
Out[52]=

$$18 - 18x - 33x^2 + 4x^3 + 5x^4$$

In[53]:=
```
df2[x_] = D[f[x], {x, 2}]
```
Out[53]=

$$-18 - 66x + 12x^2 + 20x^3$$

Nota: Observad que en la definición de df y df2, no hemos incluido asignación dinámica (usamos =, en lugar de :=). Operaremos de esta forma en todas las definiciones que involucren el cálculo de derivadas, por simplicidad. Si no lo hiciéramos así, para evaluar cualquier df[x] o df2[x] deberíamos utilizar reglas de transformación en lugar de sustitución directa.

Pasamos a calcular los puntos críticos. Intentamos primero el cálculo analítico:

In[54]:=
```
Solve[df[x] == 0, x]
```
Out[54]=

```
{{x → -2.60…}, {x → -1.10…}, {x → 0.534…}, {x → 2.36…}}
```

Tras comprobar que las expresiones de los ceros de la derivada son demasiado complicadas optamos por una aproximación numérica:

In[55]:=
```
sol = NSolve[df[x] == 0, x]
      resolvedor numérico
```

Out[55]=
```
{{x → -2.60082}, {x → -1.09686}, {x → 0.533861}, {x → 2.36382}}
```

Evaluamos este resultado en la segunda derivada. Para hacerlo, utilizaremos el operador **/.** que indica a *Mathematica* que aplique la regla de transformación que figura a continuación. Una regla de transformación es una expresión del tipo x-> y, e indica que a la variable x se le asigne el valor y. En nuestro caso:

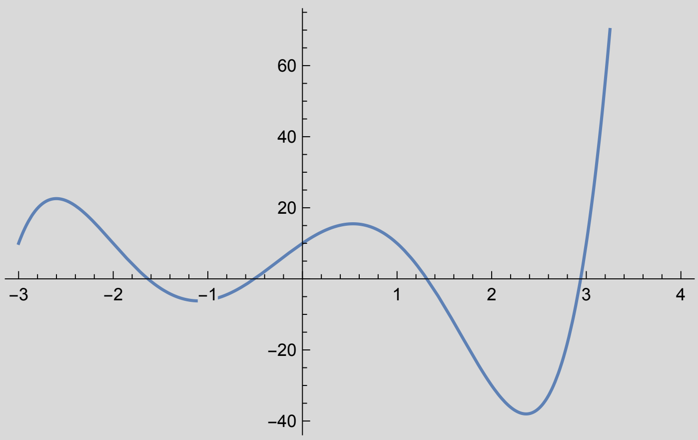
In[56]:=
```
df2[x] /. sol
```

Out[56]=
```
{-117.028, 42.4372, -46.7717, 157.202}
```

De la expresión anterior deducimos que los puntos primero y tercero son máximos y el segundo y el cuarto son mínimos. Podemos comprobarlo gráficamente:

In[57]:=
```
Plot[f[x], {x, -3, 4}]
representación gráfica
```

Out[57]=



Con respecto a los puntos de inflexión:

In[58]:=
```
sol2 = NSolve[df2[x] == 0, x]
       resolvedor numérico
```

Out[58]=
```
{{x → -2.01566}, {x → -0.26558}, {x → 1.68124}}
```

Dado que ninguno de ellos anula la tercera derivada, podemos concluir que todos son puntos de inflexión.

In[59]:=
```
df3[x_] = D[f[x], {x, 3}]
          deriva
```

Out[59]=

$$-66 + 24\,x + 60\,x^2$$

In[60]:=
```
df3[x] /. sol2
```
Out[60]=
```
{129.398, -68.142, 143.944}
```

4.3 Desarrollo de Taylor

Para el cálculo del desarrollo de Taylor en *Mathematica* disponemos del comando **Series**:

In[61]:=
```
? Series
```
Out[61]=

Symbol

Series[f, {x, x_0, n}] generates a power series expansion for f about the point $x = x_0$ to order $(x - x_0)^n$.
Series[f, {x, x_0, n_x}, {y, y_0, n_y}, …] successively finds series expansions with respect to x, then y, etc.

Aquí *f(x)* es la función, *x* la variable, *x0*, el punto donde ser realiza el desarrollo y *n* el orden.

Ejemplo. Calcular el polinomio de Taylor de grado 10 de las funciones sen(x), cos(x), exp(x) y log(1+x) en el punto x0=0

In[62]:=
```
Series[Sin[x], {x, 0, 10}]
serie  seno
```
Out[62]=
$$x - \frac{x^3}{6} + \frac{x^5}{120} - \frac{x^7}{5040} + \frac{x^9}{362\,880} + O[x]^{11}$$

In[63]:=
```
Series[Cos[x], {x, 0, 10}]
serie  coseno
```
Out[63]=
$$1 - \frac{x^2}{2} + \frac{x^4}{24} - \frac{x^6}{720} + \frac{x^8}{40\,320} - \frac{x^{10}}{3\,628\,800} + O[x]^{11}$$

In[64]:=
```
Series[Exp[x], {x, 0, 10}]
serie  exponencial
```
Out[64]=
$$1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \frac{x^5}{120} + \frac{x^6}{720} + \frac{x^7}{5040} + \frac{x^8}{40\,320} + \frac{x^9}{362\,880} + \frac{x^{10}}{3\,628\,800} + O[x]^{11}$$

In[65]:=
```
Series[Log[1 + x], {x, 0, 10}]
serie  logaritmo
```
Out[65]=
$$x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \frac{x^5}{5} - \frac{x^6}{6} + \frac{x^7}{7} - \frac{x^8}{8} + \frac{x^9}{9} - \frac{x^{10}}{10} + O[x]^{11}$$

El término O[x] que aparece al final, hacer referencia al *resto* (la diferencia que existe entre la función y el polinomio). Indica el orden del error en un entorno del punto *x0*. Si se desea trabajar solo con el polinomio, se debe anteponer el comando **Normal**, que truncará este término:

In[66]:=
```
Normal[Series[Log[1 + x], {x, 0, 10}]]
```
normal serie logaritmo

Out[66]=
$$x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \frac{x^5}{5} - \frac{x^6}{6} + \frac{x^7}{7} - \frac{x^8}{8} + \frac{x^9}{9} - \frac{x^{10}}{10}$$

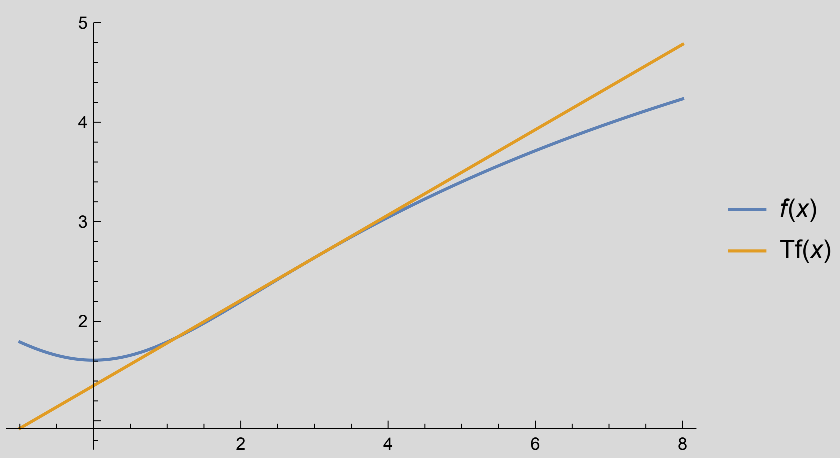
Ejemplo. Calcula la recta tangente a la curva $y = \log(x^2 + 5)$ en el punto $x = 3$.

In[67]:=
```
f[x_] := Log[x^2 + 5]
```
logaritmo

In[68]:=
```
Tf[x_] = Normal[Series[f[x], {x, 3, 1}]]
```
normal serie

Out[68]=
$$\frac{3}{7}(-3 + x) + \mathrm{Log}[2] + \mathrm{Log}[7]$$

In[69]:=
```
Plot[{f[x], Tf[x]}, {x, -1, 8}, PlotLegends → "Expressions"]
```
representación gráfica leyendas de representación

Out[69]=


Ejemplo. Calcula los polinomios de Taylor de orden 2,4,6,8,10 de la función $f(x) = 1 + \cos^2(x)$ en el punto x=0. Dibuja todas las aproximaciones con la función.

In[70]:=
```
f[x_] := 1 + Cos[x]^2;
```
coseno

In[71]:=
```
T2[x_] = Normal[Series[f[x], {x, 0, 2}]]
```
normal serie

Out[71]=
$$2 - x^2$$

In[72]:=
```
T4[x_] = Normal[Series[f[x], {x, 0, 4}]]
```
normal serie

Out[72]=
$$2 - x^2 + \frac{x^4}{3}$$

In[73]:=
```
T6[x_] = Normal[Series[f[x], {x, 0, 6}]]
```
normal serie

Out[73]=
$$2 - x^2 + \frac{x^4}{3} - \frac{2\,x^6}{45}$$

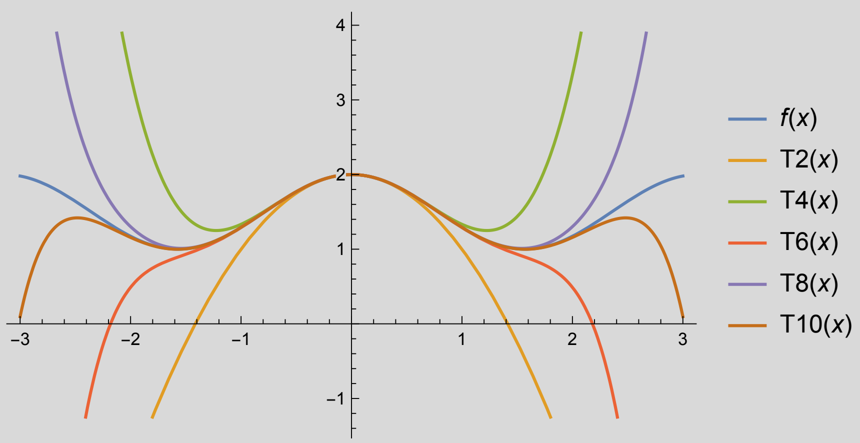
In[74]:=
```
T8[x_] = Normal[Series[f[x], {x, 0, 8}]]
```
normal serie

Out[74]=
$$2 - x^2 + \frac{x^4}{3} - \frac{2\,x^6}{45} + \frac{x^8}{315}$$

In[75]:=
```
T10[x_] = Normal[Series[f[x], {x, 0, 10}]]
```
normal serie

Out[75]=
$$2 - x^2 + \frac{x^4}{3} - \frac{2\,x^6}{45} + \frac{x^8}{315} - \frac{2\,x^{10}}{14\,175}$$

In[76]:=
```
Plot[{f[x], T2[x], T4[x], T6[x], T8[x], T10[x]},
 {x, -3, 3}, PlotLegends → "Expressions"]
```
representación gráfica
leyendas de representación

Out[76]=


In[77]:= ⋮

Out[77]= ⋮

5. Integración

5.1 Integral indefinida

Para el cálculo de primitivas con *Mathematica* disponemos del comando **Integrate**

In[78]:=
```
? Integrate
```

Out[78]=

Symbol

Integrate[f, x] gives the indefinite integral $\int f\, dx$.

Integrate[f, {x, x_{min}, x_{max}}] gives the definite integral $\int_{x_{min}}^{x_{max}} f\, dx$.

Integrate[f, {x, x_{min}, x_{max}}, {y, y_{min}, y_{max}}, …] gives the multiple integral $\int_{x_{min}}^{x_{max}} dx \int_{y_{min}}^{y_{max}} dy \ldots f$.

Integrate[f, {x, y, …} ∈ reg] integrates over the geometric region reg.

Aquí *f* es la función a integrar y *x* denota la variable de integración. Por ejemplo la primitiva de *f(x)* = x^2 sen(x)

In[79]:=
```
Integrate[x^2 Sin[x], x]
```
integra seno

Out[79]= $-\left(-2 + x^2\right) \mathrm{Cos}[x] + 2\, x\, \mathrm{Sin}[x]$

La función **Integrate** funciona aplicando reglas de transformación, de forma similar a como funciona **D**. Sin embargo, así como para la derivación existen principios generales, para la integración no existe un procedimiento sistemático de resolución. Podemos encontrarnos además con funciones para las cuales la primitiva no puede ser expresada en términos de funciones elementales. *Mathematica* resuelve dos tipos de integrales:

- Integrales cuyo integrando es combinación de funciones elementales y que pueden ser resueltas en términos de funciones elementales. Entendemos por funciones elementales: funciones racionales, exponenciales, logaritmos, funciones trigonométricas e inversas de funciones trigonométricas.
- Integrales que, por su importancia en la Economía, Estadística, Física o en otras ciencias, se expresan en términos de funciones especiales (tales como las funciones error, gamma, beta, . . .).

En general *Mathematica* resuelve todas las integrales que aparecen en las tablas de los libros. Es difícil (seguro que imposible) encontrar funciones para las que nosotros podamos calcular una primitiva y *Mathematica* no sea capaz de hacerlo. A continuación vemos unos ejemplos de cálculo de primitivas haciendo un recorrido por los tipos habituales:

- Integrales inmediatas

In[80]:=
```
Integrate[x^3, x]
```
integra

Out[80]= $\frac{x^4}{4}$

- Cambio de variable

In[81]:=
```
Integrate[(3 x^2 + 1) Exp[x^3 + x], x]
```
integra exponencial

Out[81]= e^{x+x^3}

- Integración por partes

In[82]:=
```
Integrate[Log[x], x]
```
integra logaritmo

Out[82]= $-x + x \, \mathrm{Log}[x]$

- Integración de funciones racionales

In[83]:=
```
Integrate[(x^5 + 2 x + 1) / (x^4 + 2 x^2 + 1), x]
```
integra

Out[83]= $\frac{1}{2}\left(x^2 + \frac{-3+x}{1+x^2} + \mathrm{ArcTan}[x] - 2\,\mathrm{Log}\left[1+x^2\right]\right)$

- Integrales trigonométricas

In[84]:=
```
Integrate[Sin[x]^2 Cos[x]^3, x]
```
integra seno coseno

Out[84]= $\frac{\mathrm{Sin}[x]}{8} - \frac{1}{48}\mathrm{Sin}[3\,x] - \frac{1}{80}\mathrm{Sin}[5\,x]$

- Integrales irracionales

In[85]:=
```
Integrate[Sqrt[1 - x^2], x]
```
integra raíz cuadrada

Out[85]= $\frac{1}{2}\left(x\sqrt{1-x^2} + \mathrm{ArcSin}[x]\right)$

No obstante, es conveniente recordar que existen funciones cuyas primitivas no se pueden expresar en términos de funciones elementales. En estos casos, la respuesta de *Mathematica* es la siguiente:

In[86]:=
```
Integrate[ Exp[-x^2], x]
```
integra exponencial

Out[86]= $\frac{1}{2}\sqrt{\pi}\,\mathrm{Erf}[x]$

Observar que la función da error:

In[87]:=
```
? Erf
```

Out[87]=

Symbol

Erf[z] gives the error function erf(z).

Erf[z_0, z_1] gives the generalized error function $\mathrm{erf}(z_1) - \mathrm{erf}(z_0)$.

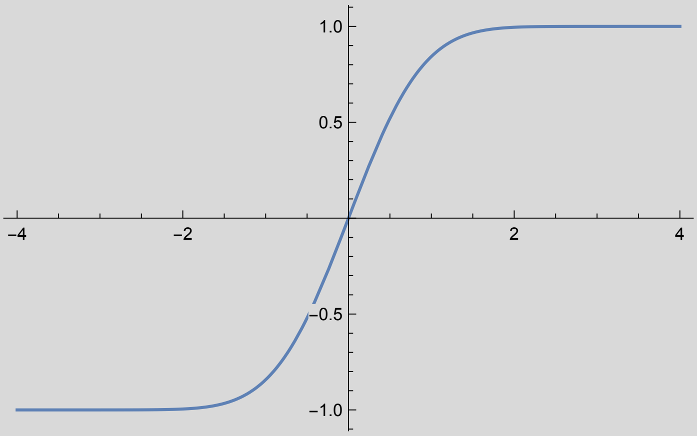
In[88]:=
```
Erf[0]
```
función error

Out[88]= 0

Su representación gráfica:

In[89]:=
```
Plot[Erf[x], {x, -4, 4}]
```
repr··· función error

Out[89]=



5.2 Integral definida

Para el cálculo de integrales definidas se utiliza también el comando **Integrate**, donde ahora además de incluir la función y variable de integración figuran los extremos de integración (véase la segunda opción de las que aparecen a continuación):

In[90]:=
```
? Integrate
```

Out[90]=

Symbol

Integrate[f, x] gives the indefinite integral $\int f\,dx$.

Integrate[f, {x, x_{min}, x_{max}}] gives the definite integral $\int_{x_{min}}^{x_{max}} f\,dx$.

Integrate[f, {x, x_{min}, x_{max}}, {y, y_{min}, y_{max}}, …] gives the multiple integral $\int_{x_{min}}^{x_{max}} dx \int_{y_{min}}^{y_{max}} dy \ldots f$.

Integrate[f, {x, y, …} ∈ reg] integrates over the geometric region reg.

Por ejemplo: $\int_0^3 (x^3 + 5x^2 - 2)\, dx$

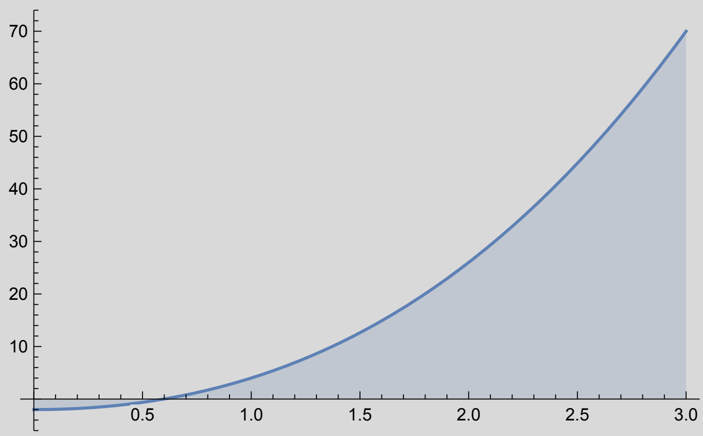
In[91]:=
```
Integrate[x^3 + 5 x^2 - 2, {x, 0, 3}]
```
integra

Out[91]= $\frac{237}{4}$

Tal y como se ha visto en teoría, la integral definida tiene una interpretación geométrica que se corresponde con el área que figura sombreada abajo (recuerda que el área que queda por debajo del eje OX se considera *negativa*)

In[92]:=
```
Plot[x^3 + 5 x^2 - 2, {x, 0, 3}, Filling → Axis]
```
representación gráfica | relleno | eje

Out[92]=



5.3 Integral impropia

El cálculo de una integral impropia combina el cálculo de primitivas y límites. Con *Mathematica* el cálculo es directo utilizando el comando **Integrate**, que admite $\pm\infty$ como extremos de integración. Es responsabilidad del usuario interpretar el resultado proporcionado por *Mathematica*. Veamos algunos ejemplos:

- Integrales impropias de primera especie (intervalo de integración no acotado)

Comprobamos que la integral $\int_1^\infty \frac{1}{x}\, dx$ es divergente:

In[93]:=
```
Integrate[1 / x, {x, 1, Infinity}]
```
integra | infinito

Integrate: Integral of $\frac{1}{x}$ does not converge on $\{1, \infty\}$.

Out[93]= $\int_1^\infty \frac{1}{x}\, dx$

En cambio $\int_1^\infty \frac{1}{x^2}\, dx$

In[94]:=
```
Integrate[1/x^2, {x, 1, Infinity}]
```
integra — infinito

Out[94]= 1

Incluso, *Mathematica,* puede trabajar con parámetros:

In[95]:=
```
Integrate[1/x^a, {x, 1, Infinity}]
```
integra — infinito

Out[95]= ConditionalExpression$\left[\frac{1}{-1+a}, \text{Re}[a] > 1\right]$

Lo cual está indicando que el valor de la integral es $\frac{1}{-1+a}$ si a es un número real mayor que 1 (en realidad basta con que su parte real sea mayor que 1, pero esto supera los contenidos de este curso). Finalizamos con otro ejemplo:

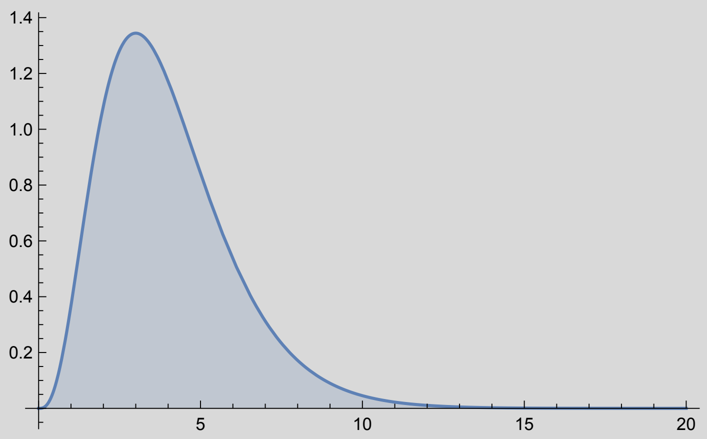
In[96]:=
```
Integrate[x^3 Exp[-x], {x, 0, Infinity}]
```
integra — exponencial — infinito

Out[96]= 6

que tiene la siguiente interpretación geométrica:

In[97]:=
```
Plot[x^3 Exp[-x], {x, 0, 20}, Filling → Axis]
```
representa··· — exponencial — relleno — eje

Out[97]=



Una integral impropia que suele aparecer en Estadística de este tipo es la conocida como función gamma:

In[98]:=
```
? Gamma
```

Out[98]=

Symbol

Gamma[z] is the Euler gamma function Γ(z).

Gamma[a, z] is the incomplete gamma function Γ(a, z).

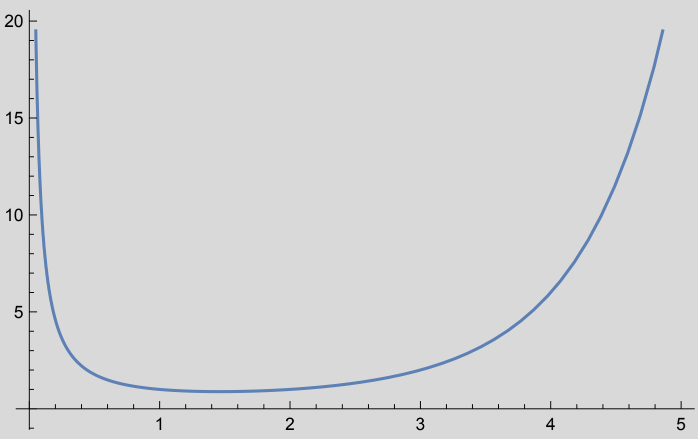
Gamma[a, z_0, z_1] is the generalized incomplete gamma function $\Gamma(a, z_0) - \Gamma(a, z_1)$.

$\Gamma(\alpha) = \int_0^\infty x^{\alpha-1} e^{-x} dx$

Si α es entero, $\Gamma(\alpha) = (\alpha\text{-}1)!$. Podemos dibujarla

In[99]:=
```
Plot[Gamma[z], {z, 0, 5}]
```
repr··· gamma de Euler

Out[99]=



O calcular algunos valores:

In[100]:=
```
Table[Gamma[n], {n, 1, 5}]
```
tabla gamma de Euler

Out[100]= {1, 1, 2, 6, 24}

Observa que obtenemos el mismo resultado integrando o utilizando la propia función de *Mathematica:*

In[101]:=
```
Integrate[x^(5/2) Exp[- x], {x, 0, Infinity}]
```
integra exponencial infinito

Out[101]= $\frac{15\sqrt{\pi}}{8}$

In[102]:=
```
Gamma[5/2 + 1]
```
gamma de Euler

Out[102]= $\frac{15\sqrt{\pi}}{8}$

Reducibles a este tipo son integrales como:

In[103]:=
```
Integrate[x^(5/3) Exp[-8 x], {x, 0, Infinity}]
integra                exponencial      infinito
```

Out[103]= $\frac{1}{256}\,\text{Gamma}\left[\frac{8}{3}\right]$

Si queremos su valor numérico:

In[104]:=
```
Integrate[x^(5/3) Exp[-8 x], {x, 0, Infinity}] // N
integra                exponencial      infinito          valor numérico
```

Out[104]= 0.00587725

- Integrales impropias de segunda especie (función no acotada en el intervalo de integración)

Corresponden a este tipo, por ejemplo:

In[105]:=
```
Integrate[1 / x, {x, 0, 1}]
integra
```

... Integrate: Integral of $\frac{1}{x}$ does not converge on {0, 1}.

Out[105]= $\int_0^1 \frac{1}{x}\,dx$

In[106]:=
```
Integrate[1/Sqrt[x], {x, 0, 1}]
integra     raíz cuadrada
```

Out[106]= 2

In[107]:=
```
Integrate[1/x^a, {x, 0, 1}]
integra
```

Out[107]= $\text{ConditionalExpression}\left[\frac{1}{1-a},\ \text{Re}[a] < 1\right]$

5.4 Integración numérica

En ocasiones no es posible integrar determinadas funciones (en realidad deberíamos decir que no es posible expresar la primitiva por medio de funciones elementales) o resulta muy costoso (en términos de tiempo) y por ello se recurre a procedimientos numéricos que se basan en la aproximación del área que delimita la curva (por tanto, esto solo es válido para integrales definidas).
La función que se encarga de este tipo de problemas en *Mathematica* es **NIntegrate** cuyos argumentos son los mismos que los de **Integrate**:

In[108]:=
```
? NIntegrate
```

Out[108]=

Symbol

NIntegrate[f, {x, x_{min}, x_{max}}] gives a numerical approximation to the integral $\int_{x_{min}}^{x_{max}} f\,dx$.

NIntegrate[f, {x, x_{min}, x_{max}}, {y, y_{min}, y_{max}}, …] gives a numerical approximation to the multiple integral $\int_{x_{min}}^{x_{max}} dx \int_{y_{min}}^{y_{max}} dy \ldots f$.

NIntegrate[f, {x, y, …} ∈ reg] integrates over the geometric region reg.

Obsérvese, por ejemplo, qué ocurre con la integral: $\int_1^3 x^3 \operatorname{sen}(x)\, e^{-x^3}\, dx$

In[109]:=
```
Integrate[x^3 Sin[x] Exp[x^(-3)], {x, 1, 3}]
```
integra seno exponencial

Out[109]= $\int_1^3 e^{\frac{1}{x^3}} x^3 \operatorname{Sin}[x]\, dx$

Sin embargo:

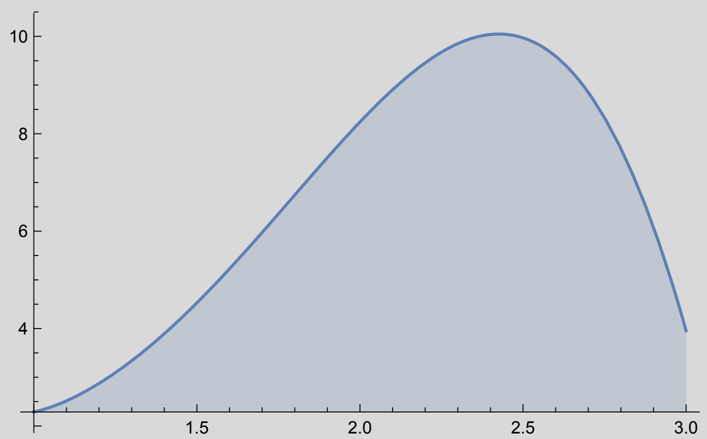
In[110]:=
```
NIntegrate[x^3 Sin[x] Exp[x^(-3)], {x, 1, 3}]
```
integra numéricame··· seno exponencial

Out[110]= 13.4607

Que es una aproximación numérica del área:

In[111]:=
```
Plot[x^3 Sin[x] Exp[x^(-3)], {x, 1, 3}, Filling → Axis]
```
representa··· seno exponencial relleno eje

Out[111]=



5.5 Aplicaciones geométricas

Una de las aplicaciones del cálculo integral es la determinación de áreas. Dadas dos curvas f, g : [a, b] → R el área que encierran sus gráficas viene dada por: $\int_a^b |\, f(x) - g(x) \,|\, dx$

Observa que la expresión anterior incluye un valor absoluto. *Mathematica* puede resolver integrales que incluyan el valor absoluto, sin embargo en general la expresión resultante va a ser compleja (porque *Mathematica* utilizará operadores lógicos para distinguir casos) y por ello es conveniente determinar la posición relativa entre las curvas f y g, y evitar la escritura del valor absoluto.

Para calcular el área entre dos curvas es importante:

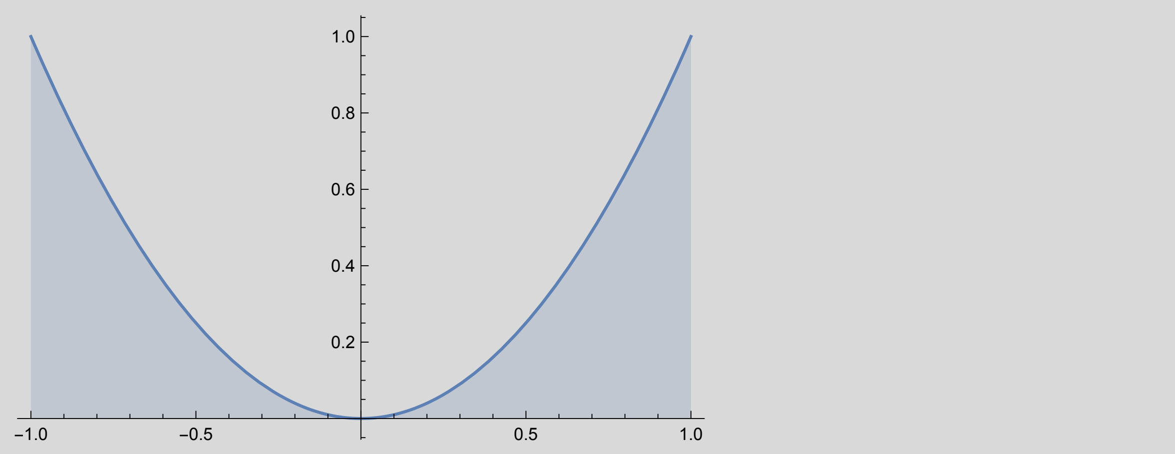
1. Conocer los puntos de intersección de las curvas, así como los cortes de éstas con el eje OX. Para estos cálculos podemos utilizar los comandos de resolución de ecuaciones.

2. Contar con un gráfico de ambas curvas

Ejemplo. Calcular el área comprendida entre la curva $y = x^2$ y el eje OX en el intervalo [-1,1]

Dibujamos la curva

In[112]:=
```
Plot[x^2, {x, -1, 1}, Filling -> Axis]
```
representación gráfica relleno eje

Out[112]=


y dado que la curva está siempre por encima del eje OX, el área es:

In[113]:=
```
Integrate[x^2, {x, -1, 1}]
```
integra

Out[113]= $\frac{2}{3}$

Ejemplo. Calcular el área de la figura delimitada por las curvas $y = \frac{1}{1+x^2}$ $y = \frac{x^2}{2}$:

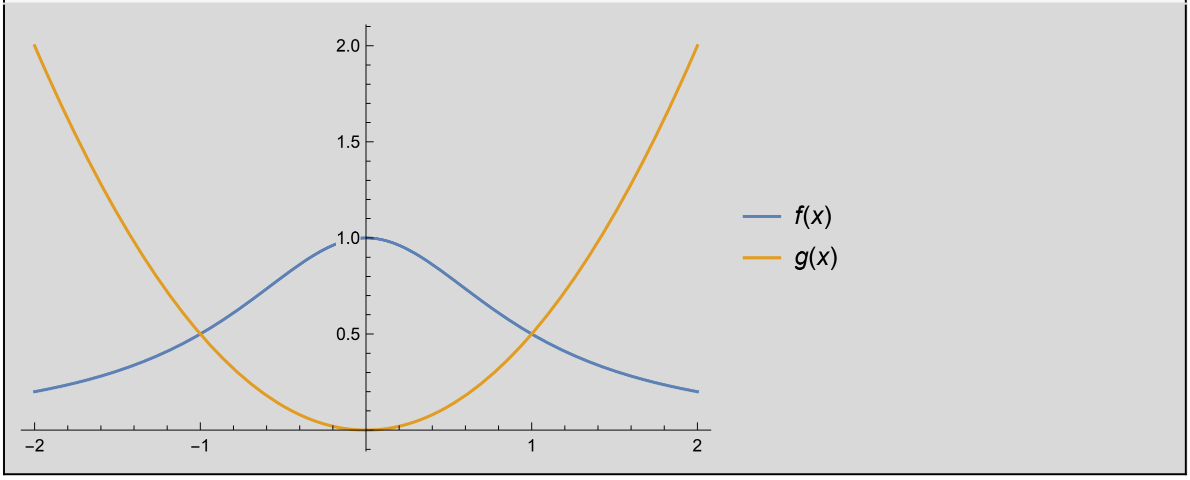
Definimos ambas funciones:

In[114]:=
```
f[x_] := 1/(1 + x^2)
```

In[115]:=
```
g[x_] := x^2/2
```

Dibujamos ambas curvas:

In[116]:=
```
Plot[{f[x], g[x]}, {x, -2, 2}, PlotLegends → "Expressions"]
```
representación gráfica   leyendas de representación

Out[116]=


Calculamos los puntos de corte:

In[117]:=
```
Solve[f[x] == g[x], x]
```
resuelve

Out[117]= $\{\{x \to -1\}, \{x \to 1\}, \{x \to -i\sqrt{2}\}, \{x \to i\sqrt{2}\}\}$

Está claro que solo nos interesan las dos soluciones reales (las dos primeras), y que en la región cuya área estamos calculando f(x) está siempre por encima de g(x), por lo tanto:

In[118]:=
```
Integrate[f[x] - g[x], {x, -1, 1}]
```
integra

Out[118]= $\frac{1}{6}(-2 + 3\pi)$

6. Ejercicios

1. Dibuja las gráficas de las siguientes funciones:
a) x(t) = t - sen(t), y(t) = 1 - cos(t), t ∈ [0, 4 π]
b) $x^{2/3} + y^{2/3} = 1$

2. La función de utilidad de un consumidor que adquiere dos bienes en cantidades x e y es u(x, y) = log xy. Dibuja las curvas de indiferencia para los valores u = 3, u = 4 y u = 10.

3. Determina si las siguientes funciones tienen simetría par o impar.
a) y = $\frac{\text{sen}(x)}{x^2+1}$
b) $y = \frac{x^3-x}{x^4+x^2+1}$
c) y = $\log x^2 + x + 1$
d) y = cos(3x) sin(x^3)

4. Dada la función y= $x^3 + 2x - 3$:
a) Dibuja su simétrica respecto del eje OX.
b) Dibuja su simétrica respecto del eje OY.
c) Dibuja la traslación de la función 3 unidades en la dirección vertical, y 2 unidades en la dirección horizontal.

5. Se supone que la evolución del precio de una acción sigue una ley:
$y(t) = t^2 - 8t + 20 + \frac{5}{t+1}\sin(10t) - \cos(3t)$
donde t representa el tiempo en días. ¿Cuál es el momento óptimo para efectuar su compra?

6. Una fábrica de plásticos recibe del Ayuntamiento de la ciudad un pedido de 8000 tablas flotadoras para el programa de natación del verano. La fábrica posee 10 máquinas, cada una de las cuales produce 50 tablas por hora. El coste de preparar las máquinas para hacer el trabajo es de 800 euros por máquina. Una vez que las máquinas están preparadas, la operación es automática y puede ser supervisada por una sola persona, que gana 35 euros/hora.
a) ¿Cuántas máquinas hay que usar para minimizar el coste de producción?
b) Si se usa el número óptimo de máquinas, ¿cuánto ganará el supervisor durante el proceso si cada tabla flotadora tiene un precio de 1,5 euros?

7. Dada la curva f(x) = $2e^x + e^{-x}$, calcula la tangente en el punto x = 1, y representa ambas gráficas en la misma figura.

8. Sea la función f(x) = log(x + 1). Calcula los polinomios de Taylor de orden 2,3,4,5 y 6 alrededor de x0 = 0 y representa sus correspondientes gráficas en una única figura.

9. Calcula las siguientes primitivas
a) $\int \frac{x^2+1}{x-1} dx$
b) $\int (2\sin(5x) - \cos(5x))\, dx$
c) $\int \log^2(x)\, dx$
d) $\int e^x \sin(x)\, dx$
e) $\int x^3\, e^{3x}\, dx$
f) $\int \frac{\sin(2x)}{1+\sin^2 x} dx$
g) $\int \frac{x^3}{\sqrt{x-1}} dx$
h) $\int \frac{x^5}{\sqrt{1-x^2}} dx$
i) $\int \frac{1}{1+e^x} dx$

10. Calcula las siguientes integrales definidas:
a) $\int_1^e \frac{1}{x} dx$
b) $\int_0^{\pi} \cos x\, dx$
c) $\int_0^5 (x^2 + 3x + 1)\, dx$
d) $\int_{-2}^2 e^{2x}\, dx$
e) $\int_1^2 \frac{1}{x+1} dx$
f) $\int_0^1 x\, e^{-3x^2}\, dx$
g) $\int_1^{\infty} e^{-3x}\, dx$
h) $\int_0^1 \frac{x}{\sqrt{x-1}} dx$

11. Si X ~ N(μ,σ) sigue una distribución normal de media μ y varianza σ, su funciones de densidad y distribución vienen dadas por:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}\, e^{\frac{-(x-\mu)^2}{2\sigma^2}}, \qquad F(x) = \int_{-\infty}^{x} f(t)\, dt$$

Suponiendo que $\mu = 10$ y $\sigma = 5$:

a) Dibuja la función de densidad.

b) Dibuja la función de distribución.

c) Comprueba que $F(\infty) = \int_{-\infty}^{\infty} f(t)\, dt = 1$.

d) Utiliza integración numérica para calcular y dibuja la función de distribución en los siguientes casos:

$P(X \leq 6) = F(6)$, $P(X \geq 3) = 1 - F(3)$, $P(X \leq 10) = F(10)$

12. Halla el área comprendida entre el eje de abscisas y la curva $y = x^3 - 6x + 8$.

13. Halla el área comprendida entre las parábolas $y = 6x - x^2$ e $y = x^2 - 2x$.