Práctica 4: Integración

En esta práctica veremos cómo se pueden calcular integrales tanto indefinidas como definidas con *Mathematica*. Además, utilizaremos esto para calcular áreas de regiones planas determinadas por funciones.

Por otro lado, cuando no disponemos del valor exacto de una integral definida existen métodos numéricos aproximados para estimar su valor. Veremos cómo usar *Mathematica* para calcular integrales definidas de forma aproximada, y en concreto, nos centraremos en el método de los Trapecios y en el de Simpson.

1. Cálculo de primitivas

Recordemos que una **primitiva** de una función $f : \mathbb{R} \to \mathbb{R}$ es otra función $F : \mathbb{R} \to \mathbb{R}$ que cumple que $F' = f$. El conjunto de todas las primitivas de f recibe el nombre de integral indefinida de f.

El comando básico de *Mathematica* para calcular **una** primitiva de una función es Integrate:

$$\textbf{Integrate}[f(x), x]$$

Otra opción es usar el símbolo que hay en la pestaña *Avanzada* de la paleta *Ayudante de clase*: $\int \blacksquare \, d\square$

∗ **Ejemplo.** Calcula la integral indefinida $\displaystyle\int \frac{\cos(\log(1+x))}{1+x}\,dx$.

Vamos a usar el comando Integrate para calcular una primitiva de $\dfrac{\cos(\log(1+x))}{1+x}$

```
In[4]:= Integrate[Cos[Log[1 + x]]/(1 + x), x]
        integra
Out[4]= Sin[Log[1 + x]]
```

Podemos comprobar que el resultado es correcto derivando la función obtenida:

```
In[5]:= D[%, x]
        deriva
Out[5]= Cos[Log[1 + x]]/(1 + x)
```

Mathematica nos devuelve una primitiva concreta de la función, pero cualquier otra primitiva se diferencia de ésta en una constante. Luego el conjunto de todas las primitivas

de esa función (o sea, su integral indefinida) será

$$\int \frac{\cos(\log(1+x))}{1+x}\,dx = \sin(\log(1+x)) + C$$

∗ **Ejemplo.** Calcula una primitiva de $e^x \sin(x)$.

```
In[1]:= Integrate[e^x Sin[x], x]
        integra      seno

Out[1]= 1/2 e^x (-Cos[x] + Sin[x])
```

Podemos comprobar que la función obtenida es una primitiva de $e^x \sin(x)$ derivándola:

```
In[2]:= D[%, x]
        deriva

Out[2]= 1/2 e^x (-Cos[x] + Sin[x]) +
        1/2 e^x (Cos[x] + Sin[x])
```

Aunque el resultado en principio no parece igual a $e^x \sin(x)$, veamos que sí lo es simplificando la expresión obtenida:

```
In[3]:= Simplify[%]
        simplifica

Out[3]= e^x Sin[x]
```

∗ **Nota.** Dada una función integrable, **no** siempre podemos obtener una expresión de una primitiva de ella en términos de funciones elementales. Por ejemplo:

```
In[4]:= Integrate[e^x^2, x]
        integra

Out[4]= 1/2 Sqrt[π] Erfi[x]
```

Mathematica devuelve el resultado en función de $\mathrm{Erfi}(x)$, la *función error imaginaria*, que es una función especial no elemental. Esta función depende a su vez de la función $\mathrm{Erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$ (*función error de Gauss*), que tiene aplicaciones en estadística y otros campos, y que no se puede expresar en términos de funciones elementales.

2. Cálculo de integrales definidas

La integral definida es un concepto utilizado para determinar el valor del área limitada por la gráfica de una función (analizaremos después esta aplicación). Si la función f es continua en un intervalo $[a, b]$, la integral definida de f en ese intervalo puede calcularse usando una primitiva de f mediante la *regla de Barrow*.

Con *Mathematica* podemos calcular de forma directa una integral definida $\int_a^b f(x)\,dx$ usando de nuevo el comando Integrate de la siguiente forma:

$$\textbf{Integrate}[f(x), \{x, a, b\}]$$

Otra opción es usar el símbolo que hay en la pestaña *Avanzada* de la paleta *Ayudante de clase*: $\int_{\square}^{\square} \blacksquare\, d\square$

∗ **Ejemplo.** Calcula la integral definida $\int_0^2 x^2\,dx$

```
In[6]:= Integrate[x^2, {x, 0, 2}]
        integra
Out[6]= 8/3
```

También podemos calcularla usando el *Ayudante de clase*:

```
In[7]:= ∫_0^2 x^2 dx
Out[7]= 8/3
```

∗ **Nota.** Como hemos visto antes, existen funciones integrables de las que no podemos obtener una expresión de una primitiva en términos de funciones elementales, como por ejemplo e^{-x^2}. Por tanto, no vamos a poder calcular una integral definida de estas funciones usando la regla de Barrow. En estos casos el comando Integrate de *Mathematica* no las resuelve, es decir, no proporciona resultados numéricos.

∗ **Ejemplo.**

```
In[9]:= Integrate[Sqrt[1 + x^3], {x, -1, 1}]
        integra
Out[9]= Sqrt[π] Gamma[4/3] / (2 Gamma[11/6]) + Hypergeometric2F1[-1/2, 1/3, 4/3, -1]
```

En estos casos, los métodos de integración numérica proporcionan la posibilidad de calcular de forma aproximada, mediante el uso de algoritmos, la integral definida de una

función. En la sección 4 veremos cómo usar con *Mathematica* dos de los métodos numéricos de integración más comunes, el método de los *Trapecios* y el método de *Simpson*.

Mathematica también dispone de un comando, NIntegrate, que usando métodos numéricos calcula una aproximación de una integral definida. La sintaxis es

NIntegrate$[f(x), \{x, a, b\}]$

Si usamos este comando podemos calcular una aproximación de la integral definida del ejemplo anterior:

```
In[10]:= NIntegrate[Sqrt[1 + x^3], {x, -1, 1}]
         integra numéricamente

Out[10]=
         1.95276
```

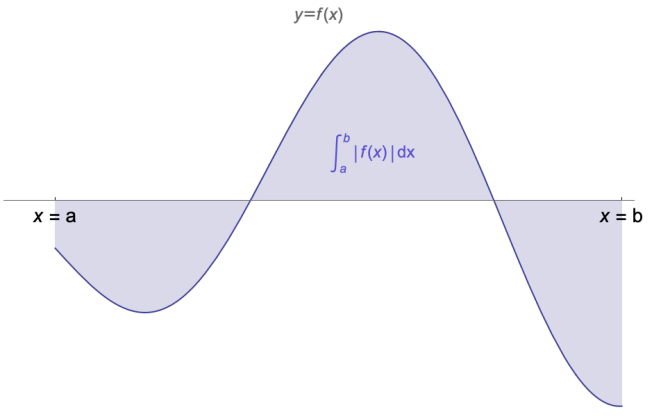
Si queremos concretar los dígitos de precisión añadiremos la opción WorkingPrecision $->$ n.

3. Cálculo de áreas

3.1. Área encerrada por la gráfica de una función

Si f es una función continua y positiva en un intervalo $[a, b]$, entonces la integral definida de f en el intervalo $[a, b]$ es el área encerrada por la curva $y = f(x)$, el eje x y las rectas $x = a$ y $x = b$. Sin embargo, si la función no es positiva en el intervalo, la integral definida no coincide con ese área. En general, si f es una función continua en el intervalo $[a, b]$, lo que se cumple es lo siguiente:

El área encerrada por la curva $y = f(x)$, el eje x y las rectas $x = a$ y $x = b$, es $\displaystyle\int_a^b |f(x)|\, dx$



Para calcular la integral definida del valor absoluto de $f(x)$, se determinarían los cambios de signo de la función f en el intervalo $[a, b]$ y se sumarían las integrales correspondientes. No obstante, esto no es necesario si estamos haciendo el cálculo con *Mathematica*, ya que el programa dispone de un comando que permite calcular el **valor absoluto**: **Abs**$[x]$

Además, podemos representar la región añadiendo en el comando Plot la opción:

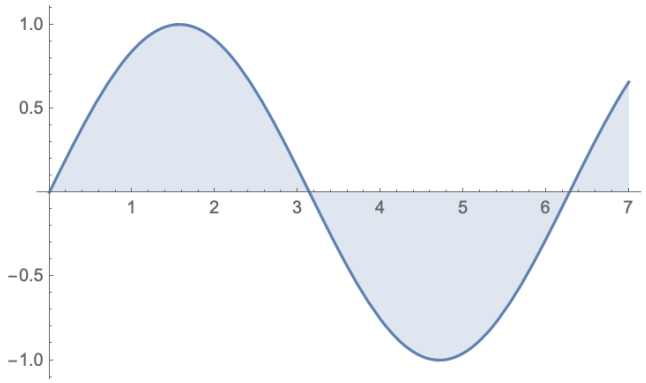
Filling $->$ Axis.

∗ **Ejemplo.** Calcula el área de la región encerrada entre la gráfica de la función $\sin(x)$ y el eje x en el intervalo $[0, 7]$.

Podemos primero representar la región:

```
In[11]:= Plot[Sin[x], {x, 0, 7}, Filling → Axis]
         repr··· seno              relleno    eje
```

Out[11]=



El valor del área de esta región se obtiene integrando el valor absoluto de la función $\sin(x)$ en el intervalo $[0, 7]$:

```
In[12]:= ∫_0^7 Abs[Sin[x]] dx
         val··· seno
```

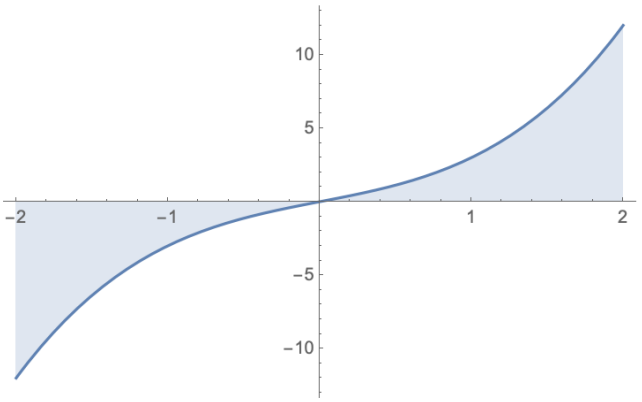
Out[12]=

```
5 - Cos[7]
```

∗ **Ejemplo.** Calcula el área de la región encerrada por la curva $y = x^3 + 2x$, el eje x y las rectas $x = -2$ y $x = 2$.

```
In[13]:= Plot[x^3 + 2 x, {x, -2, 2}, Filling → Axis]
         representación gráfica      relleno    eje
```

Out[13]=



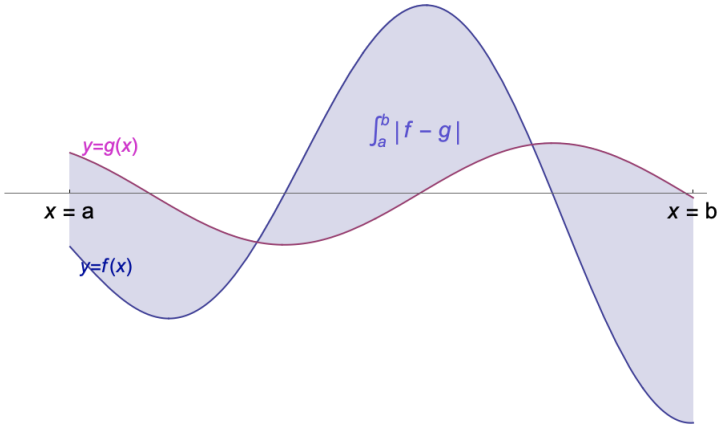
El área de esta región se obtiene integrando el valor absoluto de la función $x^3 + 2x$ en el intervalo $[-2, 2]$:

```
In[14]:= ∫_-2^2 Abs[x^3 + 2 x] dx
         valor absoluto

Out[14]=
         16
```

3.2. Área encerrada entre las gráficas de dos funciones

Si f y g son dos funciones continuas en $[a, b]$, el área encerrada entre las gráficas de las funciones $f(x)$ y $g(x)$ y las rectas $x = a$ y $x = b$ es la integral definida $\displaystyle\int_a^b |f(x) - g(x)|\, dx$



Para calcular la integral definida del valor absoluto de $f(x) - g(x)$, usaremos de nuevo el comando de *Mathematica* que permite calcular el valor absoluto, **Abs**$[x]$.

Además, podemos representar la región encerrada entre las gráficas de dos funciones añadiendo en el comando Plot la opción:

Filling −> {1 −> {2}}.

∗ **Ejemplo.** Calcula el área de la región encerrada entre las gráficas de las funciones $f(x) = 2 - x^2$ y $g(x) = x^2 - x - 2$ en el intervalo $[-2, 3]$.

Introducimos primero las funciones en *Mathematica*:

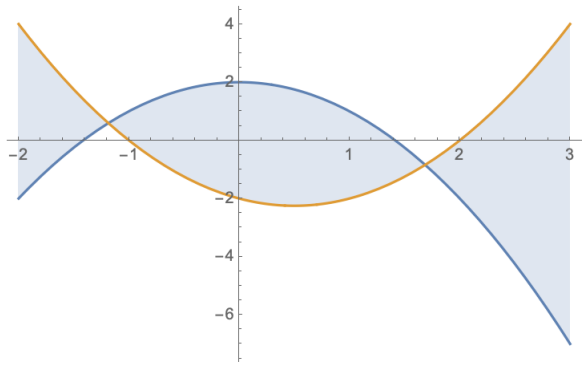
```
In[15]:= f[x_] := 2 - x^2

In[16]:= g[x_] := x^2 - x - 2
```

Antes de calcular el área, podemos representar la región en cuestión:

```
In[17]:= Plot[{f[x], g[x]}, {x, -2, 3}, Filling → {1 → {2}}]
         representación gráfica                relleno
```

Out[17]=



El área de esta región se obtiene integrando el valor absoluto de $f(x) - g(x)$ en el intervalo $[-2, 3]$:

```
In[18]:= ∫_-2^3 Abs[f[x] - g[x]] dx
         valor absoluto
```

Out[18]=

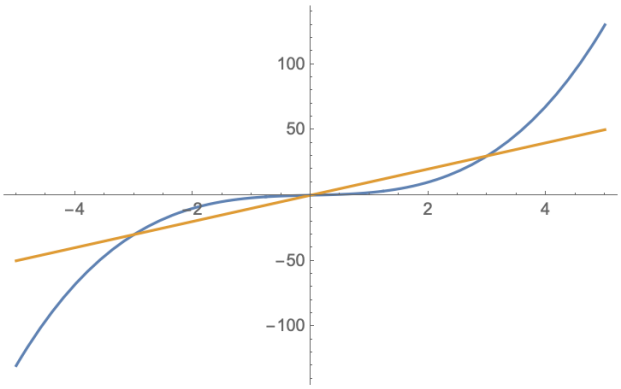
$$\frac{1}{12}\left(10 + 33\sqrt{33}\right)$$

∗ **Ejemplo.** Calcula el área encerrada entre las gráficas de las funciones $x^3 + x$ y $10x$.

En este ejemplo no nos dan el intervalo de x que delimita la región, pero si dibujamos las dos funciones vemos que delimitan una región cerrada entre sus gráficas.

```
In[19]:= Plot[{x^3 + x, 10 x}, {x, -5, 5}]
         representación gráfica
```

Out[19]=



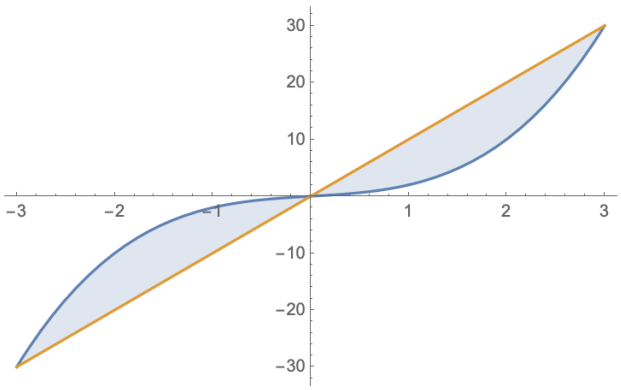
Luego, en estos casos, el valor $x = a$ será el menor punto de corte de las gráficas de las funciones, y el valor $x = b$ será el mayor punto de corte de las gráficas de las funciones.

Podemos calcular estos valores usando los comandos que vimos en la práctica 3 para resolver ecuaciones:

```
In[20]:= Solve[x^3 + x == 10 x, x]
         resuelve
Out[20]=
         {{x → -3}, {x → 0}, {x → 3}}
```

Luego la región está encerrada entre las gráficas de las dos funciones y las rectas $x = -3$ y $x = 3$.

```
In[21]:= Plot[{x^3 + x, 10 x}, {x, -3, 3}, Filling → {1 → {2}}]
         representación gráfica                relleno
Out[21]=
```



Y el área de esta región se obtiene integrando el valor absoluto de la diferencia entre $x^3 + x$ y $10x$ en el intervalo $[-3, 3]$:

```
In[22]:= ∫_-3^3 Abs[x^3 + x - 10 x] dx
         valor absoluto
Out[22]=
         81
         --
         2
```

4. Integración aproximada: Trapecios y Simpson

Los métodos de integración numérica son métodos algorítmicos que permiten calcular una aproximación del valor de la integral definida de una función cuando, por ejemplo, no es posible calcular esa integral de forma exacta mediante la regla de Barrow. En las clases de teoría se han visto dos de estos métodos: el método de los *Trapecios* y el método de *Simpson*. En esta sección vamos a estudiar como podemos utilizar *Mathematica* para aplicar estos dos métodos.

Si la integral definida que queremos aproximar es $\int_a^b f(x)\,dx$, el primer paso en ambos

métodos consiste en hacer una partición del intervalo de integración $[a, b]$:

$$P = \{a, a + h, a + 2h, \ldots, a + nh = b\},$$

donde n es el número de subintervalos y $h = \dfrac{b-a}{n}$ es la longitud de cada subintervalo.

4.1. Método de los Trapecios

Recordemos que la fórmula que permite aproximar la integral definida aplicando el Método de los Trapecios es

$$\boxed{\int_a^b f(x)\,dx \approx \frac{h}{2}\left(f(a) + 2\sum_{k=1}^{n-1} f(a+kh) + f(b)\right)}$$

Con *Mathematica* podemos construir una función que aplique la fórmula anterior:

```
Trapecios[f_, {a_, b_, n_}] := Module[{h},
                                 módulo
  h = (b - a) / n;
  N[(f[a] + 2 * Sum[f[a + k * h], {k, 1, n - 1}] + f[b]) * h / 2, 20]]
  valor numérico   suma
```

La función `Trapecios` implementa un pequeño "programa" en *Mathematica* (observad los dos puntos antes del signo =). Sus inputs son: una función f, los extremos de integración a y b, y el número de subintervalos n. Utiliza la instrucción `Module`, en la cual se declara la variable `h` como variable local, de forma que si se ha usado en la misma sesión de *Mathematica*, su valor anterior no se tiene en cuenta. El segundo argumento de la instrucción `Module` son una serie de instrucciones que efectúan cálculos con las variables locales y los inputs, separadas por ";", y finalmente la expresión que queremos calcular, en este caso, la fórmula de Trapecios.

Sabemos también que la **cota del error** cometido cuando aplicamos el método de los Trapecios es la siguiente:

$$\boxed{\frac{(b-a)^3}{12n^2} M_2} \quad \text{con } M_2 \geq \text{máx}_{[a,b]}\,|f''|$$

(es decir M_2, es una cota superior del valor absoluto de la derivada **segunda** de la función en $[a, b]$).

Podemos definir esta fórmula con *Mathematica*:

```
CotaTrapecios[a_, b_, n_, M2_] := N[((b - a) ^ 3 * M2) / (12 n ^ 2), 20]
                                  valor numérico
```

Para calcular esta cota de error, tendremos que obtener previamente M_2. Para ello representaremos el valor absoluto de la derivada segunda de f en $[a, b]$ y estimaremos gráficamente una cota superior.

En el fichero *TrapeciosSimpson.nb* tenéis el código de la función `Trapecios` y de la función `CotaTrapecios`, que podéis copiar y pegar en vuestro cuaderno de trabajo para poder utilizarlas.

∗ **Ejemplo.** Utiliza la función `Trapecios` para aproximar el valor de

$$\int_0^1 e^{-x^2}\,dx$$

dividiendo el intervalo de integración en 100 partes iguales.

Introducimos primero la función e^{-x^2} en *Mathematica* llamándola, por ejemplo, $f(x)$:

```
In[24]:= f[x_] := e^-x^2
```

Ahora podemos aplicar la función `Trapecios` para aproximar la integral de f en $[0,1]$ con 100 subdivisiones de este intervalo:

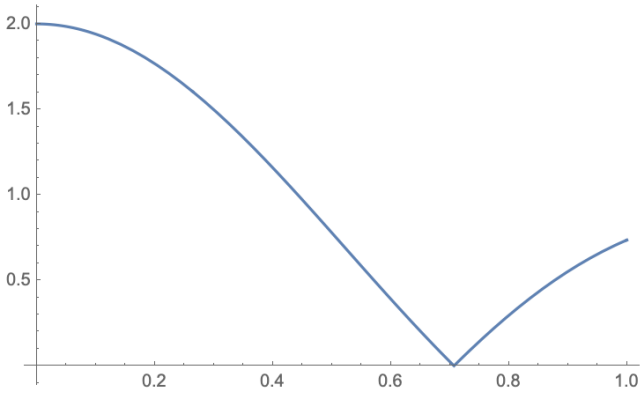
```
In[25]:= Trapecios[f, {0, 1, 100}]
Out[25]=
         0.74681800146796985197
```

Nota. El primer argumento de la función `Trapecios` es la letra que da nombre a la función (por ejemplo f) pero sin la variable (si ponemos $f[x]$ no lo entiende).

Para saber el número de decimales correctos de esta aproximación basta con que calculemos la cota de error de Trapecios en este caso. Recordad que tenemos que obtener previamente $M_2 \geq \max_{[0,1]} |f''|$. Para ello representamos el valor absoluto de la derivada segunda de f en $[0,1]$ y estimamos gráficamente una cota superior:

```
In[26]:= Plot[Abs[f''[x]], {x, 0, 1}]
         repr···  valor absoluto
```

Out[26]=



Podemos deducir del gráfico que el mayor valor de $|f''|$ en el intervalo $[0,1]$ es 2 y, por tanto, $\mathbf{M_2 = 2}$. Usando ahora la función `CotaTrapecios` podemos estimar el error cometido en la aproximación:

```
In[28]:= CotaTrapecios[0, 1, 100, 2]

Out[28]=

0.00001666666666666666667
```

Por tanto, el error cometido (es decir la diferencia entre el valor exacto de la integral definida y el valor aproximado con el método de los Trapecios con 100 subdivisiones) es menor o igual que la cota de error de Trapecios, en este caso $1{,}666666666 \cdot 10^{-5}$, que a su vez es menor que 10^{-4}. Recordad que, en general:

> Si el Error $< 10^{-p}$, la aproximación garantiza, al menos, $p-1$ decimales exactos.

Luego en este caso sabemos que la aproximación de $\int_0^1 e^{-x^2}\,dx$ obtenida con el método de los Trapecios con 100 subdivisiones tiene, al menos, 3 decimales exactos.

En efecto, si comparamos con la aproximación que nos proporciona *Mathematica* usando el comando NIntegrate,

```
In[29]:= NIntegrate[e^(-x^2), {x, 0, 1}, WorkingPrecision → 10]
         integra numéricamente        precisión operativa

Out[29]=

0.7468241328
```

vemos que la aproximación que hemos obtenido usando Trapecios con $n = 100$ (0.7468180014) tiene, en realidad, 4 decimales exactos.

∗ **Ejemplo.** Determina el número de subdivisiones a realizar en $[0, 1]$ para aproximar de nuevo la integral

$$\int_0^1 e^{-x^2}\,dx$$

con el método de los Trapecios, pero esta vez garantizando, al menos, 8 decimales exactos.

Para conseguir esta precisión, tenemos que pedir que la cota de error de Trapecios sea menor que 10^{-9}, y de ahí deduciremos el número n de subdivisiones necesarias. Recordad que para resolver desigualdades con *Mathematica* usamos el comando Reduce, y que $M_2 = 2$ en este caso (lo hemos calculado antes):

```
In[30]:= Reduce[CotaTrapecios[0, 1, n, 2] < 10^(-9), n]
         reduce

Out[30]=

n < -12 909.944487358056284 || n > 12 909.944487358056284
```

Como nos interesan los valores naturales de n, la condición relevante es que n debe ser mayor estricto que 12909, o sea, que si aplicamos el método de Trapecios con $n = 12910$ subintervalos, obtendremos una aproximación de esa integral definida con, al menos, 8 decimales exactos:

```
In[31]:= Trapecios[f, {0, 1, 12910}]
Out[31]=
0.74682413244455074789
```

Si comparamos esta aproximación de la integral con la que hemos obtenido antes usando NIntegrate (cuyos dígitos son todos correctos) vemos que la aproximación por Trapecios con $n = 12910$ tiene 9 decimales exactos, uno más de lo esperado.

4.2. Método de Simpson

La fórmula del método de Simpson que permite aproximar la integral definida es

$$\int_a^b f(x)\,dx \approx \frac{h}{3}\left(f(a) + 4\sum_{k=0}^{n/2-1} f(a+(2k+1)h) + 2\sum_{k=1}^{n/2-1} f(a+2kh) + f(b)\right)$$

Recordad que n debe ser un número par.

Podemos implementarla en *Mathematica* usando de nuevo la instrucción `Module`:

```
Simpson[f_, {a_, b_, n_}] := Module[{h},
                              módulo
  h = (b - a) / n;
  N[
  valor numérico
    (f[a] + 4 * Sum[f[a + (2 * k + 1) * h], {k, 0, n / 2 - 1}] +
                    suma
       2 * Sum[f[a + 2 * k * h], {k, 1, n / 2 - 1}] + f[b]) * h / 3, 20] ]
          suma
```

Sabemos también que la **cota del error** cometido cuando aplicamos el método de Simpson es la siguiente:

$$\frac{(b-a)^5}{180n^4}M_4 \quad \text{con } M_4 \geq \text{máx}_{[a,b]}\,|f''''|$$

(es decir M_4, es una cota superior del valor absoluto de la derivada **cuarta** de la función en $[a,b]$).

Podemos definir esta fórmula con *Mathematica*:

```
CotaSimpson[a_, b_, n_, M4_] := N[((b - a) ^ 5 * M4) / (180 n ^ 4), 20]
                                valor numérico
```

Para calcular esta cota de error tendremos que obtener previamente M_4. Para ello representaremos el valor absoluto de la derivada cuarta de f en $[a,b]$ y estimaremos gráficamente una cota superior.

De nuevo, encontraréis el código de la función `Simpson` y de la función `CotaSimpson`

en el fichero *TrapeciosSimpson.nb*. Basta con copiarlos y pegarlos en vuestro cuaderno de trabajo.

∗ **Ejemplo.** Aplica el método de Simpson para aproximar el valor de

$$\int_0^1 e^{-x^2}\,dx$$

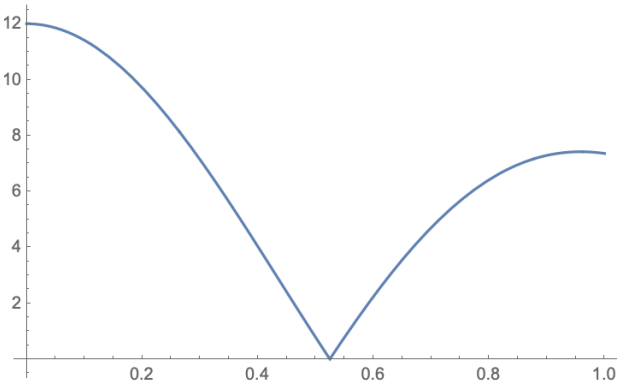
dividiendo el intervalo de integración en 100 partes iguales.

Habíamos introducido en *Mathematica* la función e^{-x^2} como $f(x)$, así que basta con utilizar la función `Simpson` para aproximar la integral de f en $[0,1]$ con 100 subdivisiones de este intervalo:

```
In[34]:= Simpson[f, {0, 1, 100}]
Out[34]=
         0.74682413289417606469
```

Para saber el número de decimales correctos de esta aproximación basta con que calculemos la cota de error de Simpson en este caso. Recordad que tenemos que obtener previamente $M_4 \geq \text{máx}_{[0,1]}\,|f''''|$. Para ello representamos el valor absoluto de la derivada cuarta de f en $[0,1]$ y estimamos gráficamente una cota superior:

```
In[35]:= Plot[Abs[f''''[x]], {x, 0, 1}]
         repr··· valor absoluto
Out[35]=
```



Deducimos que el mayor valor de $|f''''|$ en el intervalo $[0,1]$ es 12 y, por tanto, $\mathbf{M_4 = 12}$. Calculamos la cota de error de Simpson para estimar el error cometido en la aproximación:

```
In[36]:= CotaSimpson[0, 1, 100, 12]
Out[36]=
         6.6666666666666667 × 10^-10
```

Por tanto, el error cometido es menor o igual que la cota de error de Simpson que, en este caso, es menor que 10^{-9}. Luego la aproximación de $\int_0^1 e^{-x^2}\,dx$ usando el método de Simpson con 100 subdivisiones tiene, al menos, 8 decimales exactos.

En efecto, si comparamos con la aproximación de la integral que habíamos obtenido en el ejemplo anterior usando el comando NIntegrate, vemos que la aproximación que hemos obtenido usando Simpson con $n = 100$ tiene, en realidad, 10 decimales correctos.

Nota. Observa que para conseguir 8 decimales exactos mediante el método de los Trapecios han hecho falta 12910 subdivisiones frente a las 100 del método de Simpson.

∗ **Ejemplo.** Determina ahora el número de subdivisiones a realizar en $[0, 1]$ para aproximar de nuevo la integral

$$\int_0^1 e^{-x^2}\, dx$$

con el método de Simpson, pero esta vez garantizando, al menos, 15 decimales exactos.

Para conseguir esta precisión, tenemos que pedir que la cota de error de Simpson sea menor que 10^{-16}, y de ahí vamos a deducir el número n de subdivisiones necesarias. Recordad que $M_4 = 12$ en este caso (lo hemos calculado antes):

```
In[39]:= Reduce[CotaSimpson[0, 1, n, 12] < 10^(-16), n]
         reduce
Out[39]=
         n < -5081.3274815461473628 || n > 5081.3274815461473628
```

Como nos interesan los valores naturales de n, la condición relevante es que n debe ser mayor estricto que 5081, y recordad que en el caso del método de Simpson n debe ser par, así que si aplicamos el método de Simpson con $n = 5082$ subintervalos obtendremos una aproximación de esa integral definida con, al menos, 15 decimales exactos:

```
In[40]:= Simpson[f, {0, 1, 5082}]
Out[40]=
         0.74682413281242703766
```

Si comparamos con la aproximación que nos proporciona *Mathematica* usando el comando NIntegrate con 20 dígitos de precisión:

```
In[41]:= NIntegrate[e^(-x^2), {x, 0, 1}, WorkingPrecision → 20]
         integra numéricamente          precisión operativa
Out[41]=
         0.74682413281242702540
```

vemos que la aproximación por Simpson con $n = 5082$ tiene 16 decimales exactos, uno más de lo esperado.