# Mathematica: potentes herramientas

Carlos L. Arce Salas
Escuela de Matemática
Universidad de Costa Rica

**Introducción**

Aún si su trabajo parece no estar vinculado con la matemática, Mathematica puede ser de su interés. Con este recurso el arduo trabajo del cálculo --numérico o simbólico- resulta cosa del pasado, el desarrollo de materiales didácticos tiene nuevas y revolucionarias herramientas, las aplicaciones de modelos matemáticos pueden producir resultados sin ocuparse de la implementación computacional de complicados algoritmos matemáticos, en suma, con las computadoras y Mathematica se multiplican las capacidades para entender, desarrollar y aplicar las matemáticas y ciencias afines.

El optimismo de las ideas expuestas, sin embargo, no tiene la intención de hacer pensar que en Mathematica se encontrarán soluciones mágicas a problemas como el aprendizaje de la matemática, por ejemplo, o al de sus aplicaciones. Pero sí, enfatizar que Mathematica es una herramienta que puede ser redefinida y aplicada a gran diversidad de problemas, y en variadas formas.

---



---

## ¿Qué es mathematica?

Todo lenguaje computacional hace del computador una herramienta de propósito general, pero entre estos *Mathematica* es uno muy especial, por varias razones.

Sin disponer de un dominio completo del paquete, en las primeras sesiones de trabajo, es posible obtener la solución a gran cantidad de problemas de la matemática y de sus aplicaciones. Un extenso conjunto de elaborados métodos pueden ser aplicados directamente, sacando provecho de la herramienta sin necesidad de ocuparse de ella como lenguaje de programación. Observe por ejemplo, las ilustraciones presentadas en las siguientes secciones.

Pero aunque *Mathematica* dispone de gran cantidad de procedimientos ya definidos, en todos los campos de la matemática y sus aplicaciones, adquiere su verdadera potencia cuando permite utilizarlos para redefinir y crear nuevos procedimientos. En este aspecto agrega a los tradicionales recursos de los lenguajes de programación, renovados recursos para hacer cálculo numérico y la construcción de gráficos y nuevas herramientas para hacer cálculo simbólico, creando un nuevo paradigma de

programación: la programación simbólica.

Con todo ello *Mathematica* se convierte en una extensión del lenguaje de la matemática, que permite no solo representar y transformar información o conocimientos, si no que le da a la matemática una nueva dimensión al posibilitar el ensayo y la exploración, como medio habitual de aproximarse a los problemas y su solución.

A continuación se presentan algunos ejemplos de aplicación del paquete, con la intención de ilustrar las apreciaciones dadas, pero que sólo representan a una breve introducción al amplio mundo de *Mathematica*. Para obtener una visión más completa de los alcances del paquete, se recomienda visitar su página web: www.wolfram.com/mathematica.

## Editor de texto especializado

En la primera aproximación a *Mathematica* es posible reconocer un editor de texto científico, que permite hacer cálculos numéricos y algebraicos integrando también la construcción de gráficos.

Descargar CGauss.nb

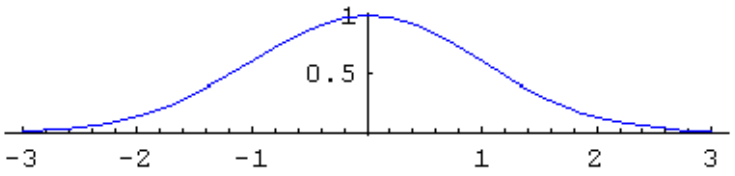
CGauss.nb

Cálculo del área bajo la curva $y = e^{-x^2/2}$ en el intervalo $[0, \infty[$.

```
Needs["Graphics`Colors`"];
Needs["Graphics`FilledPlot`"]
```

```
f[x_] := e^(-x^2/2)
```

```
g1 = Plot[f[x], {x, -3, 3}, AspectRatio → Automatic,
   Ticks → {Automatic, {0.5, 1}}, PlotStyle → {Blue}];
```




Para cada t, el área bajo la curva $y = f(x)$, entre 0 y t, es dada por:

```
F[t_] := ∫_0^t f[x] dx
```

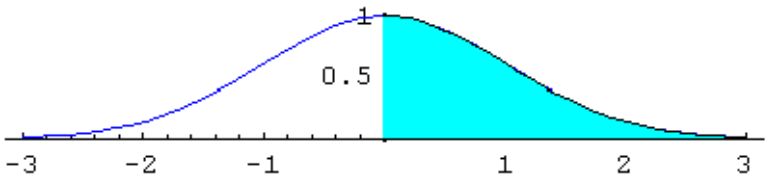
```
Limit[F[t], t → ∞]
```

$$\sqrt{\frac{\pi}{2}}$$

Con el último comando se verifica que $\int_0^{\infty} e^{-x^2/2}\,dx = \sqrt{\pi/2}$.

```
g2 = FilledPlot[f[x], {x, 0, 3}, Fills → {Cyan}];
```

```
Show[g1, g2];
```

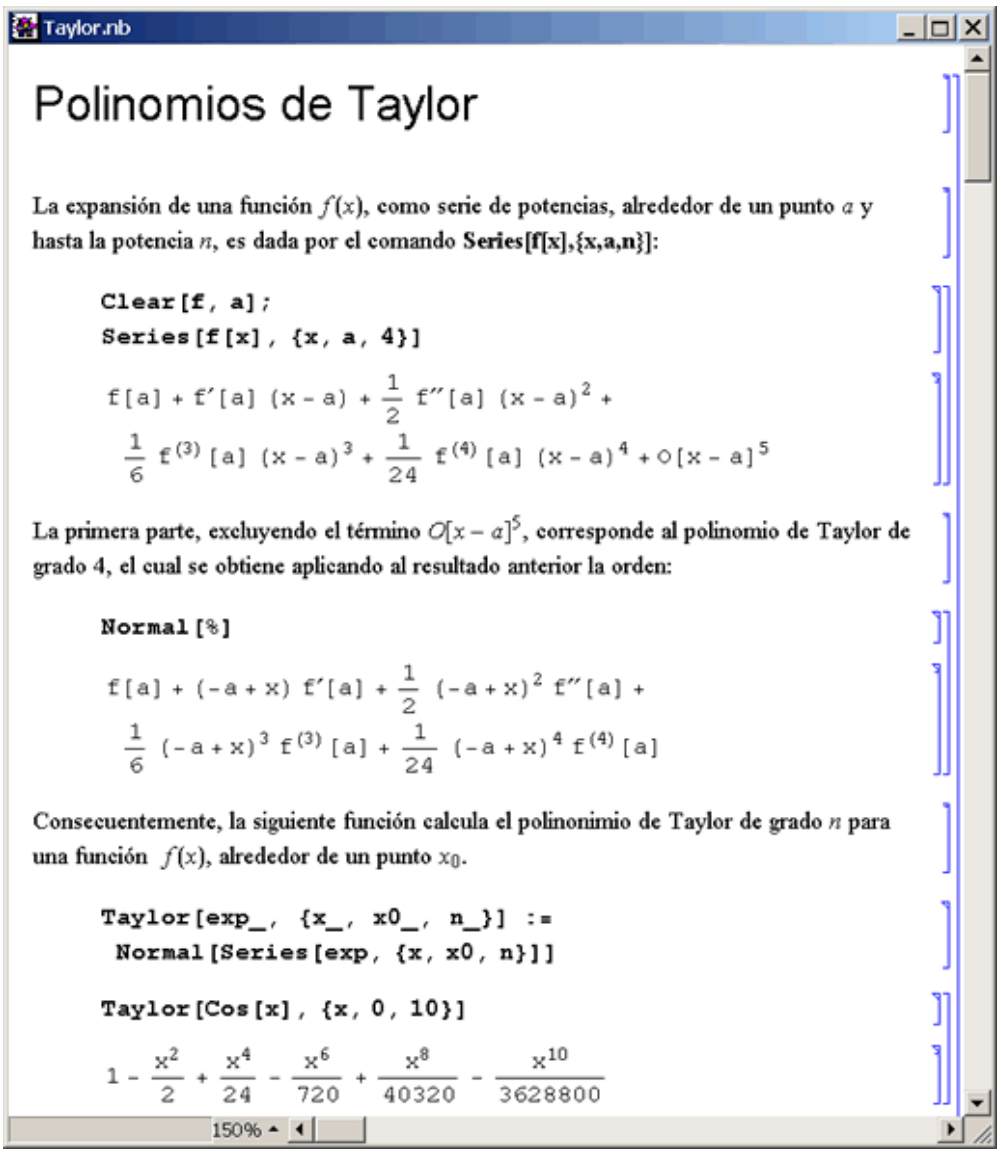



150%

Este editor se presenta como una interface entre *Mathematica* y el usuario, denominada FrontEnd, que permite dar órdenes a *Mathematica* para que realice las tareas requeridas, y que además ofrece excelentes recursos para documentar el trabajo realizado.

Para el proceso de enseñanza y aprendizaje de la matemática, los documentos creados con el FrontEnd pueden tener la calidad tipográfica de un libro de texto, pero con un texto ``vivo'' que permite ensayar y explorar los diversos conceptos de la matemática y las ciencias. Así las computadoras se convierten en interesantes laboratorios, que abren posibilidades para que el aprendizaje de la matemática sea un proceso más experimental y con experiencias más significativas para el estudiante.

Descargar Taylor.nb

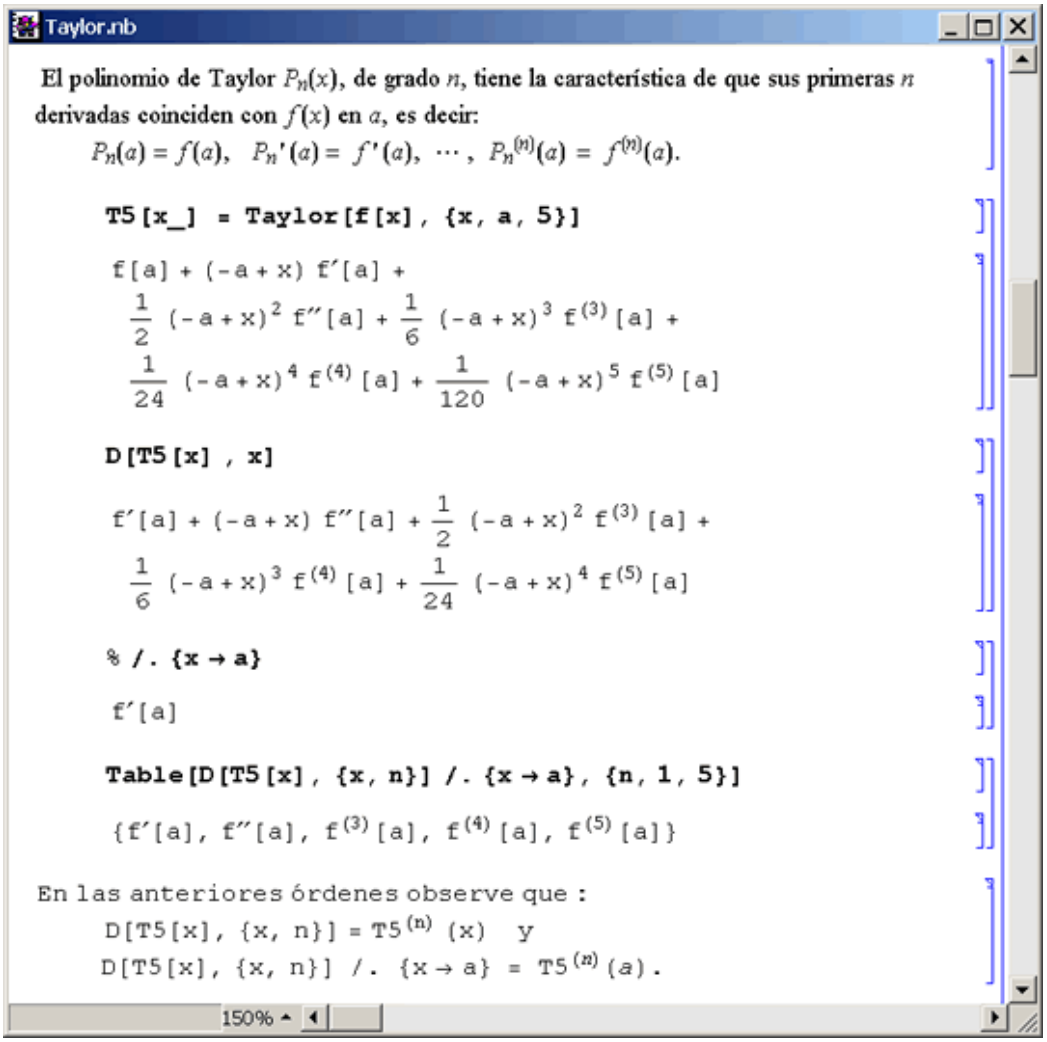



La versión 4.2 de *Mathematica* agrega a los tradicionales recursos de este editor, la biblioteca de herramientas Author tools que, entre otros aspectos, permite definir tablas de contenidos e índices de materia, con vínculos apropiados a los contenidos del documento, para dar acabados de libro o artículo a los documentos producidos con el FrontEnd.

## Algebra simbólica

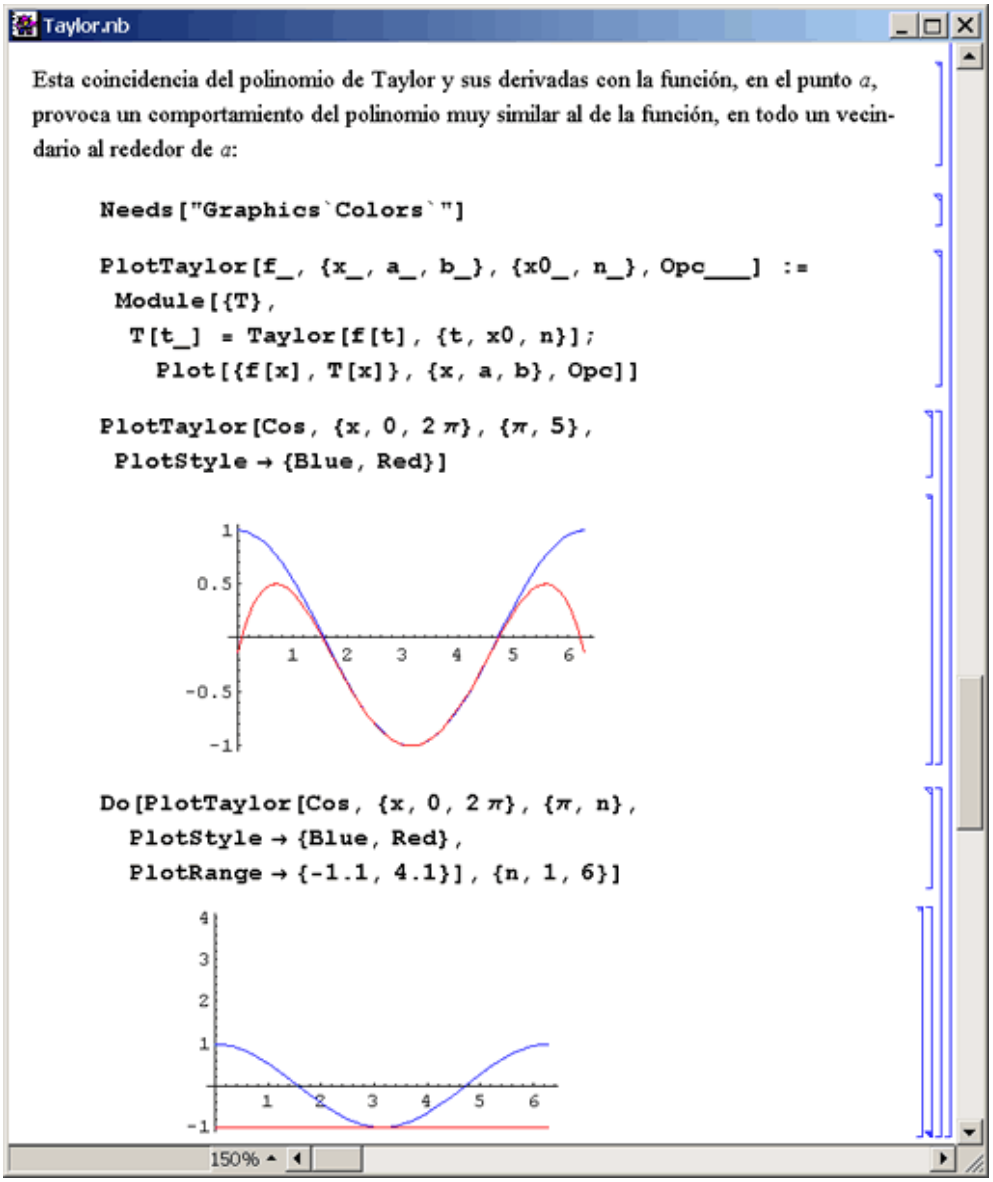
Cuando se observa a *Mathematica* por su capacidad para efectuar cálculo simbólico, se encuentra un amplio y diverso conjunto de procedimientos para trabajar en virtualmente todos los temas de la matemática. Cálculo aritmético exacto, búsqueda de fórmulas cerradas para raíces de ecuaciones polinomiales, diferenciación e integración simbólica, sólo para citar unos pocos aspectos de un punto fuerte de *Mathematica*.

Descargar Taylor.nb




En el ejemplo expuesto, se observa la habilidad de *Mathematica* para trabajar con una función $f(x)$ de manera simbólica, al calcular los polinomios de Taylor de grado $n$, alrededor de $a$. También se hace notar que, aunque *Mathematica* aporta muchos métodos, el usuario siempre necesitará hacer adaptaciones o definir nuevos procedimientos, para adecuar la exploración a sus necesidades.

Descargar Taylor.nb




En la ilustración anterior se define el procedimiento `PlotTaylor`, que extiende el comando `Plot` de *Mathematica* para construir, simultáneamente, el gráfico de $f(x)$ y el de su respectivo polinomio de Taylor, de grado $n$ y alrededor de $x_0$. Preservando en la extensión todos los parámetros opcionales de `Plot`. Con este comando se construyen 6 gráficos --de los que es visible sólo el primero-- para producir una animación en la que se muestra como al incrementar el grado del polinomio de Taylor, este aproxima mejor la función en un intervalo mayor alrededor de $x_0$.

## Mathematica como lenguaje de programación

*Mathematica* soporta los clásicos comandos If, Do, For, Wile para el control de ciclos, que caracterizan los tradicionales lenguajes de programación, permitiendo escribir programas al estilo de Pascal y C por ejemplo. Pero este es solo un aspecto secundario del potente lenguaje de programación que constituye *Mathematica*. La nueva forma de programación creada por *Mathematica* incluye, además, la programación al estilo de LisP o Scheme, con las funciones Map y Apply, funciones puras, y la recursividad entre otros recursos, que dan a *Mathematica* un marcado estilo de programación funcional. Y a todo ello se agregan las herramientas de programación del álgebra simbólica, patrones y reglas de reescritura, creando un nuevo paradigma de programación: la programación simbólica.

Este estilo de programación se caracteriza por usar una forma unificada para representar todos los tipo de elementos que

involucra: datos, listas, fórmulas, gráficos, funciones, programas, etc. *Mathematica* acude al concepto de **expresión** para este propósito: una **expresión** es un objeto con estructura de árbol que pueden escribirse como f[x1,...,xn], entendiendo que cada parámetro xi es otra expresión, y en la que los números y filas de caracteres son las expresiones elementales o átomos. Como consecuencia de esta forma de representación, por ejemplo, no se hacen mayores diferencias entre datos y programas, de manera que resulta natural que una función reciba funciones como parámetros y también pueda producir una función como resultado. Con las expresiones para representar todo tipo de elementos, se ofrecen diversidad de recursos para operar con ellas, entre los cuales se distinguen los patrones y reglas de reescritura. En la siguiente sección se propone un ejemplo que utiliza estos elementos.

En resumen, *Mathematica* unifica los nuevos recursos de la programación simbólica con los de la programación funcional, para lograr un lenguaje que permite escribir algoritmos muy concisos y poderosos. En particular, es posible escribir programas que generan o modifican otros programas. Aunado a esto, la enorme cantidad de métodos que aporta *Mathematica* y sus bibliotecas, convierten a este paquete en una potente plataforma de programación.

## Patrones y reglas de reescritura

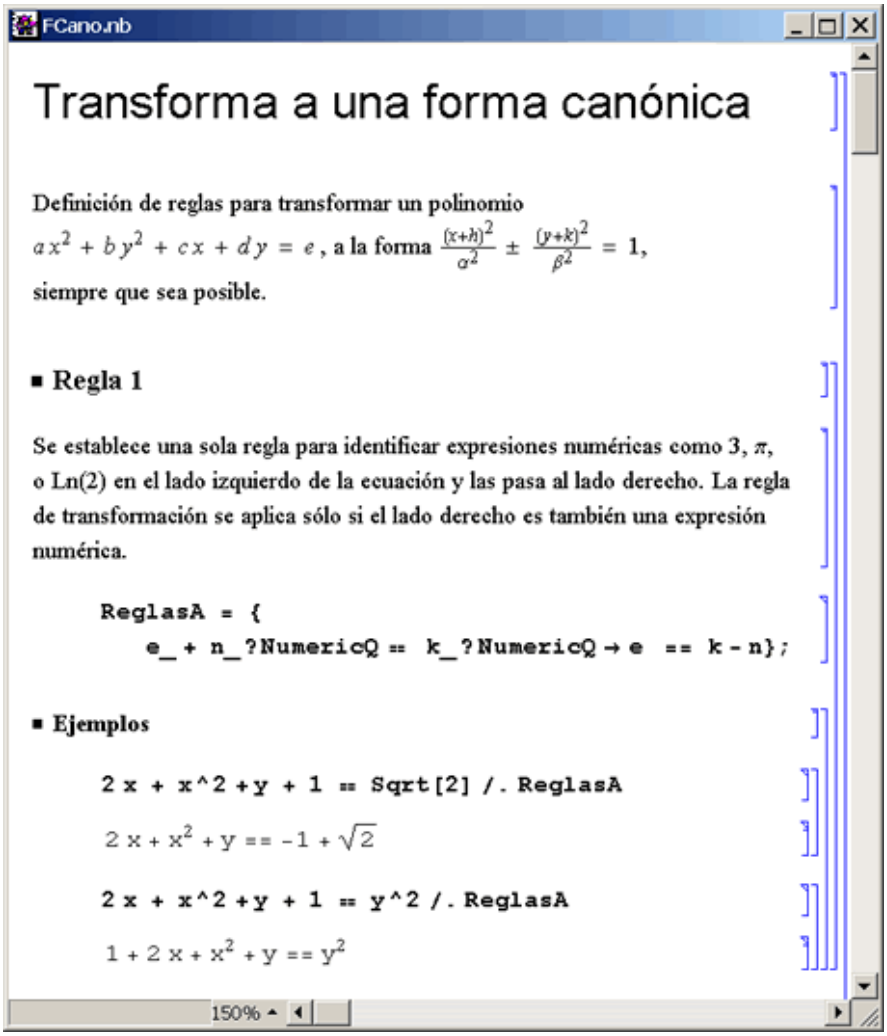
Con la finalidad de presentar un ejemplo de programación simbólica, a continuación se discute el uso de patrones y reglas de reescritura para definir un procedimiento que determina la forma canónica de la ecuación de una sección cónica, a partir de su ecuación polinomial.

La idea general del método es proponer un conjunto de reglas, cuya aplicación permita transformar la ecuación polinómica en dos variables, a la forma canónica de una cónica. Para simplificar el problema sólo se consideran las formas canónicas de elipses e hipérbolas:

$$\frac{(x-h)^2}{a^2} \pm \frac{(y-k)^2}{b^2} = 1.$$

Los patrones en *Mathematica* son expresiones que incluyen el objeto `Blank[ ]` identificado usualmente con el símbolo de subrayado (_ ), y que representa a cualquier tipo de expresión --cualquier cosa en *Mathematica*--. Así por ejemplo: `x_` es un patrón que se asocia o empata con cualquier expresión y se denomina `x`, `n_Integer` es un patrón para una expresión denominada `n` que tiene cabeza `Integer`, es decir que es un entero y `x_Symbol^2` representa cualquier símbolo elevado al cuadrado.

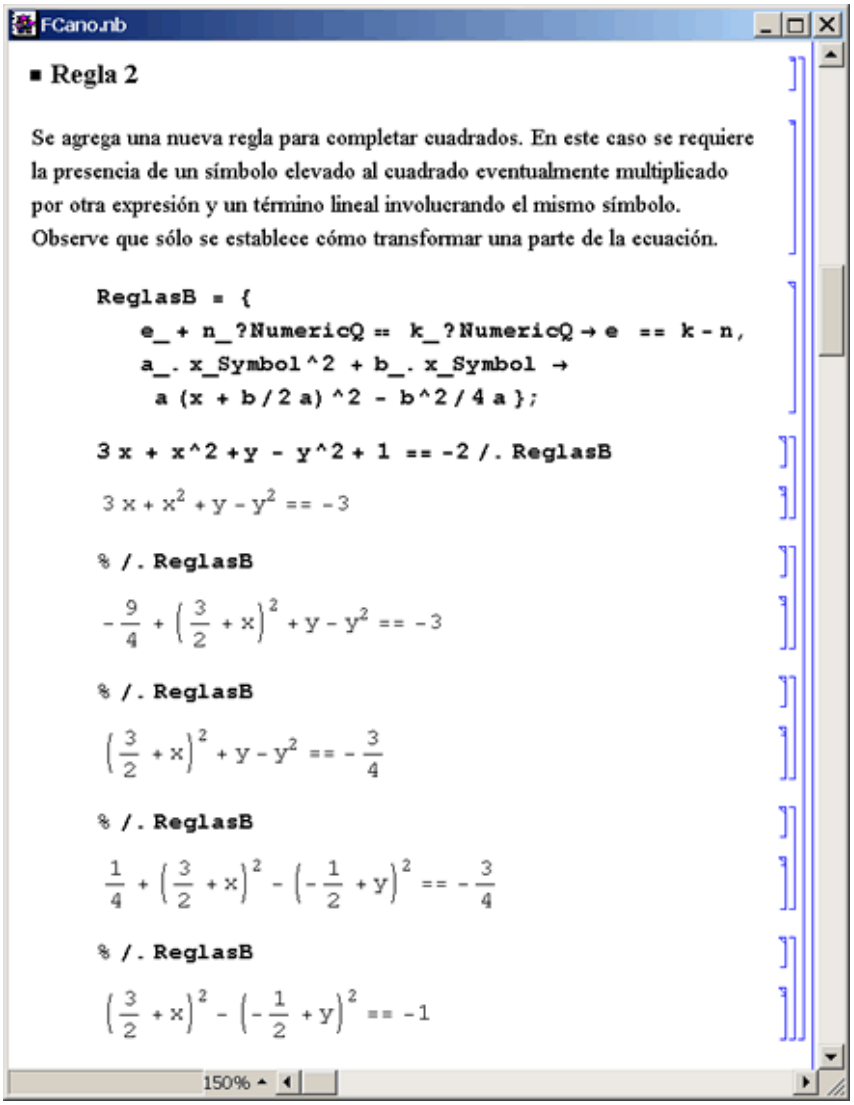
Descargar FCano.nb




Una regla de transformación es una expresión de la forma `patrón -> expresión`, que utiliza un `patrón` para establecer la forma de las expresiones a las que se aplicará y una `expresión` --en términos de los elementos del `patrón`-- que determina la nueva forma que debe asumir expresión. Para aplicar las reglas se usa el operador `/.` denominado `Replace`.

Por ejemplo, en la ilustración anterior, se da una regla para transformar ecuaciones pasando constantes numéricas al lado derecho. En este caso, cuando se aplica la regla a la primera ecuación, el patrón `e_` empata con la expresión $2x + x^2 + y$. Y cuando se aplica a la segunda ecuación no se provoca ningún cambio, porque la ecuación no empata con el patrón de la regla -- el lado derecho no es una expresión numérica -- y por lo tanto la transformación no se aplica.

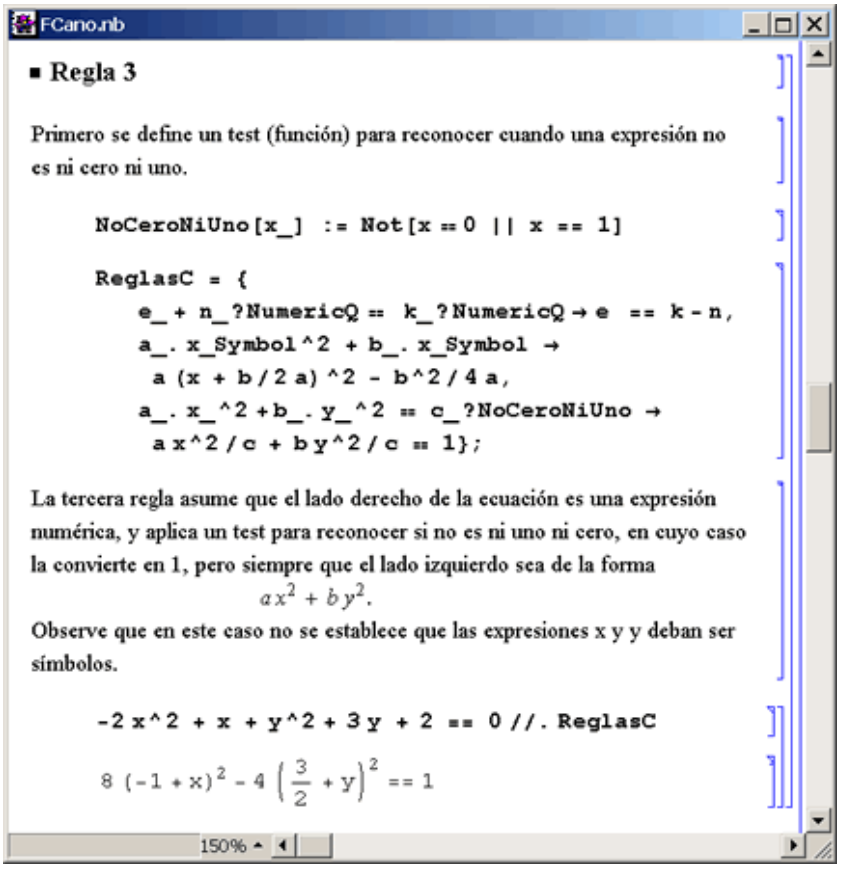
Descargar FCano.nb




Con la regla dos se usa el patrón `a_. x_Symbol^2` que empata con cualquier producto de una expresión denominada `a` con un símbolo al cuadrado, por ejemplo $3z^2$. El punto en el patrón se utiliza para que un solo símbolo al cuadrado también empate con el patrón, asumiendo en ese caso que `a = 1`. Observe que el patrón `x_^ 2` puede empatar con expresiones como $t^2$, $(y - 1)^2$ o $(\cos(x))^2$, pero `x_Symbol^ 2` empata sólo si el término al cuadrado es un símbolo como $t^2$, $z^2$ o $x^2$.

Por otra parte, en la siguiente ilustración, se usa el operador `//. -ReplaceAll-`, para aplicar las reglas recursivamente hasta que ninguna regla sea aplicable al resultado obtenido.

Descargar FCano.nb




La definición de una función en *Mathematica*, por ejemplo `f[x_] := x^2+1`, es una regla de transformación global que se aplica en todo lugar donde la expresión `f[x]` tenga lugar. Así, a continuación se definen dos funciones o reglas de transformación globales, identificadas con el nombre `aFormaCanonica`. La primera, `aFormaCanonica[liz_ == lde_]`, recibe una ecuación en cualquier forma, pasa a la izquierda su parte derecha, dejando en cero esta última. Luego le aplica las reglas de transformación vistas con anterioridad y que han sido registradas como una lista denominada `Reglas`. Finalmente se invoca recursivamente a sí misma, pero asegurándose de que el nuevo argumento sea de la forma

$$ax^2 + by^2 = 1,$$

para que se aplica la segunda regla global para darle finalmente la forma:

$$\frac{x^2}{\alpha^2} + \frac{x^2}{\beta^2} = 1.$$

Naturalmente, la regla `aFormaCanonica[a_.x_^2 + b_.y_^2 == 1]` dada se aplica sólo cuando el argumento sea una ecuación que empate con el patrón dado. *Mathematica* ordena las reglas globales según su grado de generalidad, de lo particular a lo más general, y se aplican en ese orden.

Descargar FCano.nb


Aunque los ejemplos dados tienen un perfil introductorio, todos ellos han querido mostrar que *Mathematica* provee cantidad de procedimientos para obtener soluciones directas a gran diversidad de problemas matemáticos y de las ciencias en general. Y que el lenguaje para hacerlo es también un lenguaje de programación con enorme potencial. Un lenguaje de programación, que comienza a desbordar los sistema del álgebra simbólica, para encontrar aplicaciones en otros campos de las ciencias de la computación.

## Novedades de la versión 4.2

Los ejemplos presentados hasta ahora, corren en las versiones 4.0 y 4.1, pues sólo incluyen elementos de *Mathematica* ya consolidados en las versiones anteriores.

La versión 4.2 de *Mathematica* presenta varias novedades, entre otras, los procedimientos de programación lineal resultan ahora mejores y de uso más natural, también se ofrecen refinamientos en los métodos de optimización, en estadística se agrega el nuevo paquete ANOVA y, en combinatoria y teoría de grafos el paquete Combinatorica. Pero tal vez, el aspecto más novedoso incluido con la nueva versión es la incorporación de MathLink al núcleo de *Mathematica*. Con ello se logra, de manera transparente y sin necesidad de ningún tipo de instalación, una conexión de dos vías con JAVA, (J/link 2.0), que permite desde *Mathematica* crear objetos JAVA e invocar sus métodos y desde JAVA usar y controlar el núcleo de *Mathematica*. Sin duda, otra interesante extensión del lenguaje de *Mathematica* que habilita para usar los extensos recursos de JAVA.

En el notebook JavaEjem.nb se ilustra este nuevo recurso, presentando un ejemplo en el que se usan desde *Mathematica* varias clases de la biblioteca AWT de JAVA, con el propósito de crear un selector del ViewPoint, para un gráfico en 3

dimensiones ``en vivo'', a fin de establecer el punto desde donde se desea observarlo. Los detalles de estos programas requieren consultar el manual en línea que aporta *Mathematica*: J/Link User Guide.

---