Manual de KAlgebra

Aleix Pol
Traductor: Leticia Martín Hernández
Traductor: Eloy Cuadra

Índice general

1. Introducción **7**

2. Sintaxis **9**

3. Uso de la calculadora **11**

4. Gráficas 2D **13**
4.1. Sintaxis . . . 13
4.2. Características . . . 13

5. Gráficas 3D **15**

6. Diccionario **17**

7. Órdenes permitidas por KAlgebra **18**
7.1. más . . . 18
7.2. veces . . . 18
7.3. menos . . . 18
7.4. dividir . . . 18
7.5. cociente . . . 19
7.6. potencia . . . 19
7.7. radicación . . . 19
7.8. factorial . . . 19
7.9. y . . . 19
7.10. o . . . 20
7.11. xor . . . 20
7.12. negación . . . 20
7.13. mcd . . . 20
7.14. mcm . . . 20
7.15. resto . . . 21
7.16. factorof . . . 21
7.17. max . . . 21
7.18. min . . . 21
7.19. lt . . . 21
7.20. gt . . . 22
7.21. eq . . . 22

7.22. neq . . . 22
7.23. leq . . . 22
7.24. geq . . . 22
7.25. implies . . . 23
7.26. approx . . . 23
7.27. abs . . . 23
7.28. floor . . . 23
7.29. ceiling . . . 23
7.30. sin . . . 24
7.31. cos . . . 24
7.32. tan . . . 24
7.33. sec . . . 24
7.34. csc . . . 24
7.35. cot . . . 24
7.36. sinh . . . 25
7.37. cosh . . . 25
7.38. tanh . . . 25
7.39. sech . . . 25
7.40. csch . . . 25
7.41. coth . . . 25
7.42. arcsin . . . 26
7.43. arccos . . . 26
7.44. arctan . . . 26
7.45. arccot . . . 26
7.46. arccosh . . . 26
7.47. arccsc . . . 26
7.48. arccsch . . . 27
7.49. arcsec . . . 27
7.50. arcsech . . . 27
7.51. arcsinh . . . 27
7.52. arctanh . . . 27
7.53. exp . . . 28
7.54. ln . . . 28
7.55. log . . . 28
7.56. conjugate . . . 28
7.57. arg . . . 28
7.58. real . . . 29
7.59. imaginary . . . 29
7.60. sum . . . 29
7.61. product . . . 29
7.62. diff . . . 29
7.63. card . . . 30
7.64. scalarproduct . . . 30
7.65. selector . . . 30

7.66. union 30
7.67. forall 30
7.68. exists 31
7.69. map 31
7.70. filter 31
7.71. transponer 31

8. Créditos y licencia **32**

Resumen

KAlgebra es una aplicación que puede sustituir su calculadora gráfica. Posee funciones numéricas, lógicas, simbólicas y de análisis que le permiten calcular expresiones matemáticas en la calculadora y trazar gráficamente el resultado en 2D o 3D. KAlgebra está arraigada al lenguaje de marcas matemáticas (MathML); no obstante, no es necesario saber MathML para usar KAlgebra.

Capítulo 1

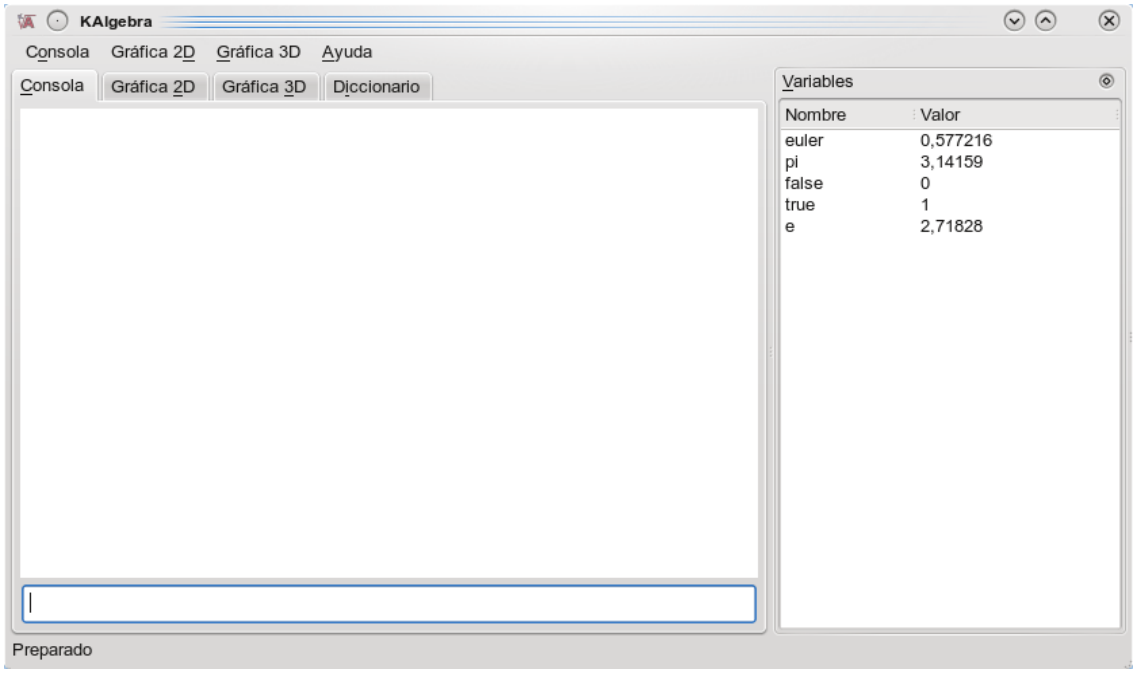
Introducción

KAlgebra contiene numerosas funciones que permiten al usuario realizar todo tipo de operaciones matemáticas y mostrarlas gráficamente. Originalmente, este programa estaba orientado a MathML. En la actualidad lo puede usar cualquier persona con pocos conocimientos matemáticos para resolver problemas sencillos o avanzados.

Incluye funcionalidades como las siguientes:

- Una calculadora para una fácil y rápida evaluación de funciones matemáticas.
- Posibilidad de usar guiones para series avanzadas de cálculos.
- Posibilidades lingüísticas, como la definición de funciones y la terminación automática.
- Funciones de cálculo, como la diferenciación simbólica, cálculo vectorial y manipulación de listas.
- Trazado de funciones con cursor activo para encontrar raíces gráficas y otros tipos de análisis.
- Trazado en 3D para la visualización útil de funciones en 3D.
- Un diccionario de operadores integrado para la rápida referencia de las diversas funciones disponibles.

A continuación se muestra una imagen de la aplicación KAlgebra en acción:



Cuando el usuario comienza una sesión con KAlgebra se le muestra una única ventana que contiene cuatro pestañas: **Calculadora**, **Gráfica 2D**, **Gráfica 3D** y **Diccionario**. Dentro de estas pestañas encontrará un campo de entrada donde podrá introducir sus funciones o cálculos, y un campo de visualización que muestra el resultado.

El usuario puede gestionar la sesión en cualquier momento con las opciones del menú principal **Sesión**:

Sesión → Nueva (Ctrl+N)

Abre una nueva ventana de KAlgebra

Sesión → Modo de pantalla completa (Ctrl+Mayúsculas+F)

Conmuta el modo de pantalla completa para la ventana de KAlgebra. El modo de pantalla completa también se puede activar o desactivar mediante el botón situado en la parte superior derecha de la ventana de KAlgebra.

Sesión → Salir (Ctrl+Q)

Cierra el programa.

Capítulo 2

Sintaxis

KAlgebra usa una sintaxis algebraica intuitiva para introducir las funciones del usuario de forma similar a la que usan la mayor parte de las calculadoras gráficas modernas. Esta sección lista las operaciones fundamentales que proporciona KAlgebra. El autor de KAlgebra ha modelado la sintaxis según Maxima y Maple para los usuarios que estén familiarizados con estos programas.

Para los usuarios interesados en el funcionamiento interno de KAlgebra, las funciones introducidas por el usuario se convierten a MathML en el motor. Un conocimiento rudimentario de las posibilidades proporcionadas por MathML le servirá de ayuda para revelar las funcionalidades internas de KAlgebra.

Aquí tiene una lista de los operadores disponibles hasta el momento:

- `+ - * /` : suma, resta, multiplicación y división.
- `^, **` : potencia, puede usar cualquiera de las dos formas. Asimismo, puede usar los caracteres unicode ². Las potencias son también una forma de expresar raíces, lo que puede hacer de la siguiente forma: `a**(1/b)`
- `->` : lambda. Es el modo de especificar una o más variables libres que estarán ligadas a una función. Por ejemplo, en la expresión **`length:=(x,y)->(x*x+y*y)^0.5`**, el operador lambda se usa para indicar que `x` e `y` estarán ligadas cuando se usa la función «length».
- `x=a..b` : se usa cuando necesitamos delimitar un intervalo (variable limitada + límite superior + límite inferior). Esto significa que `x` va de `a` a `b`.
- `()` : se usa para especificar una prioridad mayor.
- `abc(parámetros)` : funciones. Cuando el analizador sintáctico encuentra una función, comprueba si `abc` es un operador. Si lo es, lo trata como un operador; si no lo es, lo trata como una función de usuario.
- `:=` : definición. Se usa para definir el valor de una variable. Puede escribir cosas como **`x:=3`**, **`x:=y`**, donde `y` puede estar definido o no, o como **`perímetro:=r->2*pi*r`**.
- `?` : definición de condiciones en funciones definidas a trozos («piecewise»). Permite definir operaciones condicionales en KAlgebra. Dicho de otro modo, es otra forma de especificar una condición «if, elseif, else». Cuando introducimos la condición antes del símbolo `?`, se utilizará dicha condición solo si es verdadera; en cambio, si encuentra un símbolo `?` sin ninguna condición, se tendrá en cuenta el último caso. Por ejemplo: **`piecewise { x=0 ? 0, x=1 ? x+1, ? x**2 }`**
- `{ }` : contenedor MathML. Puede usarse para definir un contenedor. Es muy útil a la hora de trabajar con funciones definidas a trozos.
- `= > >= < <=` : operadores relacionales que indican «igual», «mayor», «mayor o igual», «menor» o «menor o igual», respectivamente.

Ahora puede preguntarme por qué debería el usuario preocuparse por MathML. La respuesta es fácil. Con esto podremos realizar operaciones con funciones como `cos()`, `sin()` (o cualquier otra función trigonométrica), `sum()` o `product()`. No importa el tipo de función que sea. Podremos usar `plus()`, `times()` y cualquier cosa a la que le corresponda un operador. También se han implementado funciones lógicas, por lo que podremos hacer cosas como `or(1,0,0,0,0)`.

Capítulo 3

Uso de la calculadora

La calculadora de KAlgebra es útil como una calculadora a lo grande. El usuario puede introducir expresiones a evaluar en los modos **Calcular** y **Evaluar**, según la selección del menú **Calculadora**.

En el modo de evaluación, KAlgebra simplifica la expresión incluso si se encuentra una variable indefinida. Cuando está en modo de cálculo, KAlgebra calcula todo y si encuentra una variable indefinida muestra un error.

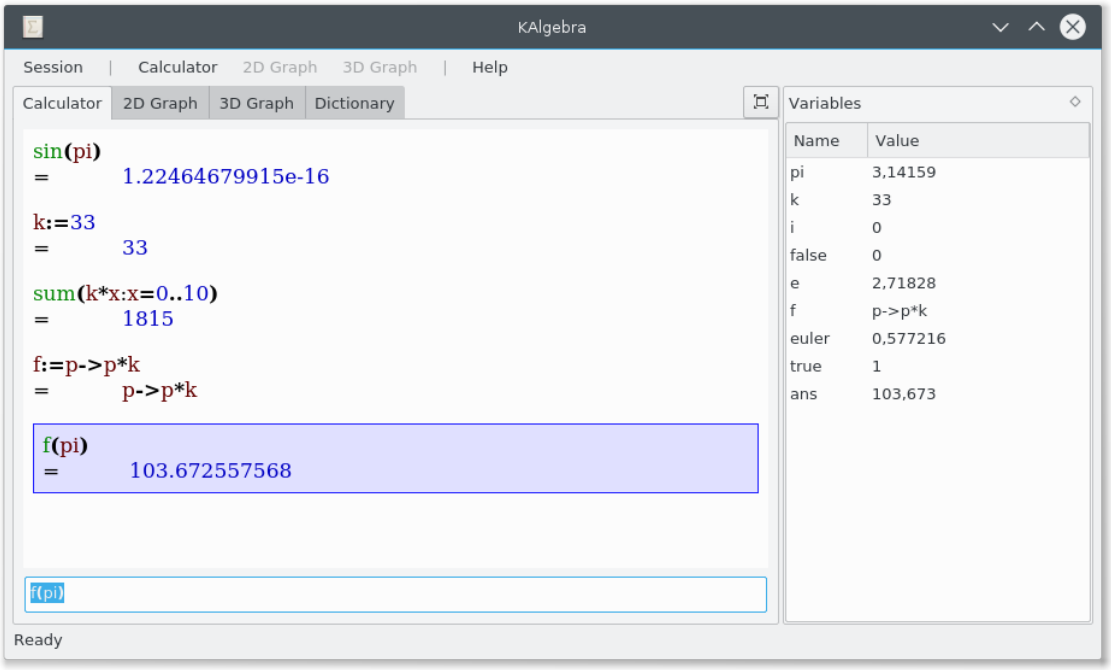
Además de mostrar las ecuaciones introducidas por el usuario y el resultado en la pantalla de la calculadora, también se muestran todas las variables declaradas en un cuadro persistente en la parte de la derecha. Si hace doble clic sobre una variable se mostrará un diálogo que le permitirá modificar su valor (solo es un modo de engañar al registro).

La variable «`ans`» es especial; cada vez que introduzca una expresión, la variable «`ans`» cambiará su valor al del último resultado.

A continuación se muestran funciones de ejemplo que se pueden introducir en el campo de entrada de la ventana de la calculadora:

- **`sin(pi)`**
- **`k:=33`**
- **`sum(k*x : x=0..10)`**
- **`f:=p->p*k`**
- **`f(pi)`**

A continuación se muestra una captura de la ventana de la calculadora tras introducir las expresiones de ejemplo anteriores:



El usuario puede controlar la ejecución de una serie de cálculos usando las opciones del menú **Calculadora**:

Calculadora → Cargar guion… (Ctrl+L)

Ejecuta las instrucciones de un archivo de forma secuencial. Es útil si desea definir bibliotecas o reanudar un trabajo anterior.

Calculadora → Guiones recientes

Muestra un submenú que le permitirá escoger los guiones usados recientemente.

Calculadora → Guardar guion… (Ctrl+G)

Guarda las instrucciones que ha introducido desde que comenzó la sesión, con lo que podrá reutilizarlas. Genera un archivo de texto que podrá corregir usando cualquier editor de texto, por ejemplo Kate.

Calculadora → Exportar registro… (Ctrl+S)

Guarda el registro con todos los resultados en un archivo HTML que podrá imprimir o publicar.

Calculadora → Insertar ans… (F3)

Insertar la variable «`ans`» y facilitar la reutilización de valores anteriores.

Calculadora → Calcular

Un botón de opción para definir el **Modo de ejecución** a cálculo.

Calculadora → Evaluar

Un botón de opción para definir el **Modo de ejecución** a evaluación.

Capítulo 4

Gráficas 2D

Para añadir un gráfica 2D en KAlgebra, seleccione la pestaña **Gráfica 2D** y pulse la pestaña **Añadir**, que le permitirá añadir una nueva función. A continuación tendrá dirigirse al cuadro de texto de entrada, donde podrá introducir la función.

4.1. Sintaxis

Si quiere usar una función `f(x)` típica, no es necesario que la especifique; en cambio, si quiere usar una función `f(y)` o una función polar, tendrá que añadir `y->` y `q->` como variables limitadas.

Ejemplos:

- **`sin(x)`**
- **`x²`**
- **`y->sin(y)`**
- **`q->3*sin(7*q)`**
- **`t->vector{sin t, t**2}`**

Después de haber introducido la función, pulse el botón **Aceptar** para dibujar la gráfica en la ventana principal.

4.2. Características

Es posible dibujar varias gráficas en la misma vista. Para esto use simplemente el botón **Añadir** cuando se encuentre en el modo Lista. Puede ajustar el color de cada gráfica.

Puede ampliar la vista y desplazarla con el ratón. Con la rueda del ratón podrá ampliar o reducir la vista. Puede también seleccionar un área con el botón izquierdo del ratón, y el área se ampliará. Desplace la vista con las teclas de dirección del teclado.

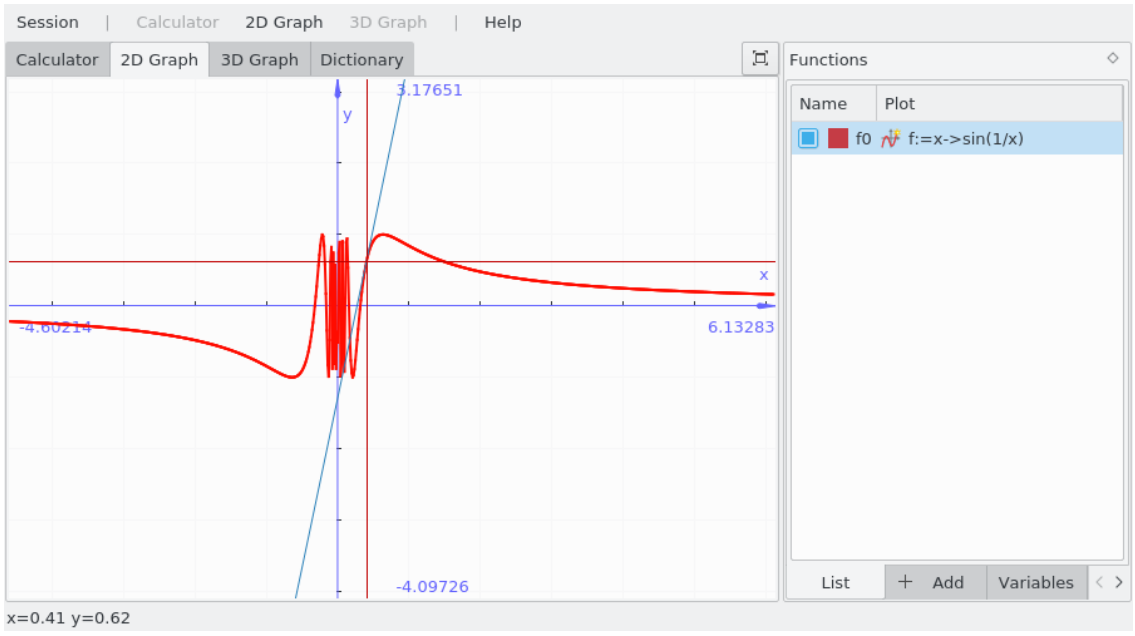
> NOTA
> La ventana de gráficos 2D se puede definir de manera explícita mediante la pestaña **Ventana** en una pestaña **Gráfico 2D**.

En la pestaña **Lista** de la parte inferior derecha puede abrir una pestaña de **Edición** para editar o eliminar una función mediante doble clic y poner o quitar la marca de la casilla de verificación situada junto al nombre de la función para mostrarla u ocultarla.

En el menú **Gráfica 2D** dispone de las siguientes opciones:

- **Cuadrícula**: Mostrar u ocultar la cuadrícula.
- **Mantener las proporciones**: Mantener las proporciones mientras amplía o reduce la vista.
- **Guardar**: Guardar (**Ctrl**+**S**) la gráfica como un archivo de imagen.
- **Ampliar/reducir**: Ampliar (**Ctrl**+**+**) y reducir (**Ctrl**+**-**).
- **Tamaño real**: Reiniciar la vista a la ampliación original.
- **Resolución**: Seguida por una lista de botones de opciones para seleccionar una resolución para los gráficos.

A continuación se muestra una captura de pantalla de un usuario cuyo cursor está en la raíz del extremo derecho de la función, `sen(1/x)`. El usuario ha utilizado una resolución muy alta para mostrar el grafo (ya que oscila al frecuencias muy altas cerca del origen). También dispone de la funcionalidad de cursor vivo, mediante la cual se muestran los valores `x` e `y` en la esquina inferior izquierda de la pantalla cada vez que mueva el cursor a un nuevo punto. Se traza una «línea tangente» viva sobre la función en la posición del cursor vivo.



Capítulo 5

Gráficas 3D

Para dibujar una grafica 3D con KAlgebra, seleccione la pestaña **Gráfica 3D**, donde verá un campo de entrada en la parte inferior que le permitirá introducir la función. Aún no podrá definir `z`. Por ahora, KAlgebra solo admite gráficas 3D que dependen de `x` e `y`, como **`(x,y)->x*y`**, donde **`z=x*y`**.

Ejemplos:

- **`(x,y)->sin(x)*sin(y)`**
- **`(x,y)->x/y`**

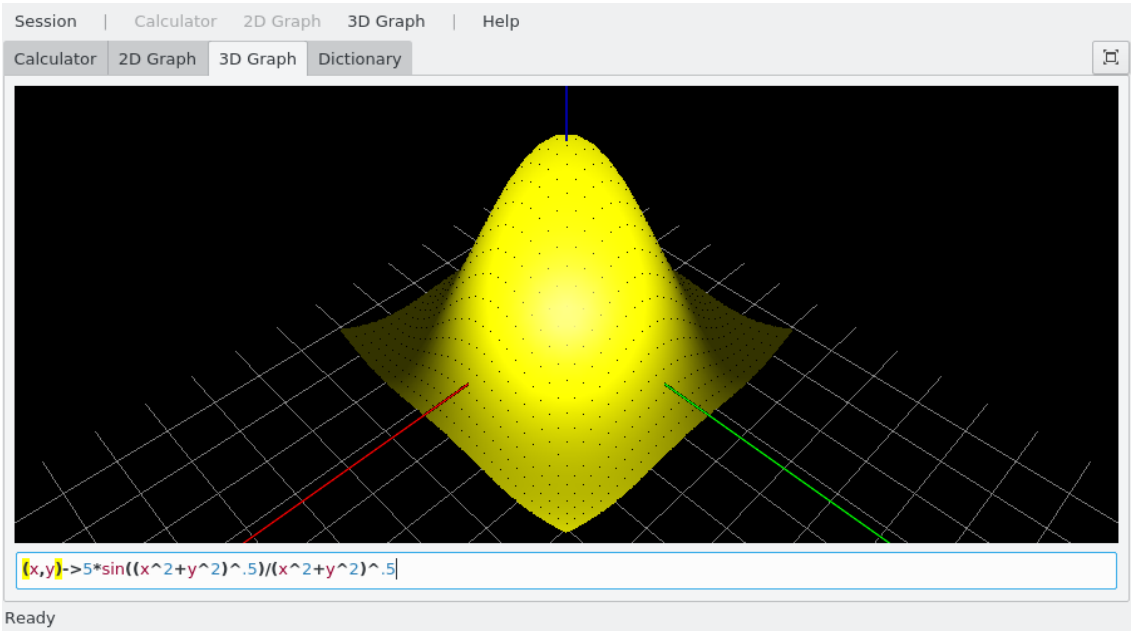
Puede ampliar o reducir la vista con el ratón. Use la rueda del ratón para ampliarla o reducirla. Si mantiene pulsado el botón izquierdo del ratón y se desplaza con el ratón, la gráfica rotará.

Las teclas de las flechas **Flecha izquierda** y **Flecha derecha** rotan el grafo alrededor del eje `z`. Las teclas de las flechas **Flecha arriba** y **Flecha abajo** rotan la vista alrededor del eje horizontal. Pulse **W** para ampliar el gráfico y **S** para reducirlo.

En el menú **Gráfica 3D** dispone de las siguientes opciones:

- **Guardar**: Guardar (**Ctrl**+**S**) la gráfica como un archivo de imagen o un documento permitido.
- **Reiniciar la vista**: Reiniciar la vista a la ampliación original en el menú **Gráfico 3D**.
- Puede dibujar los gráficos con los estilos de **Puntos**, **Líneas** o **Sólido** en el menú **Gráfico 3D**.

A continuación se muestra una captura de la función denominada «sombrero». Esta gráfica particular se muestra usando el estilo 3D.

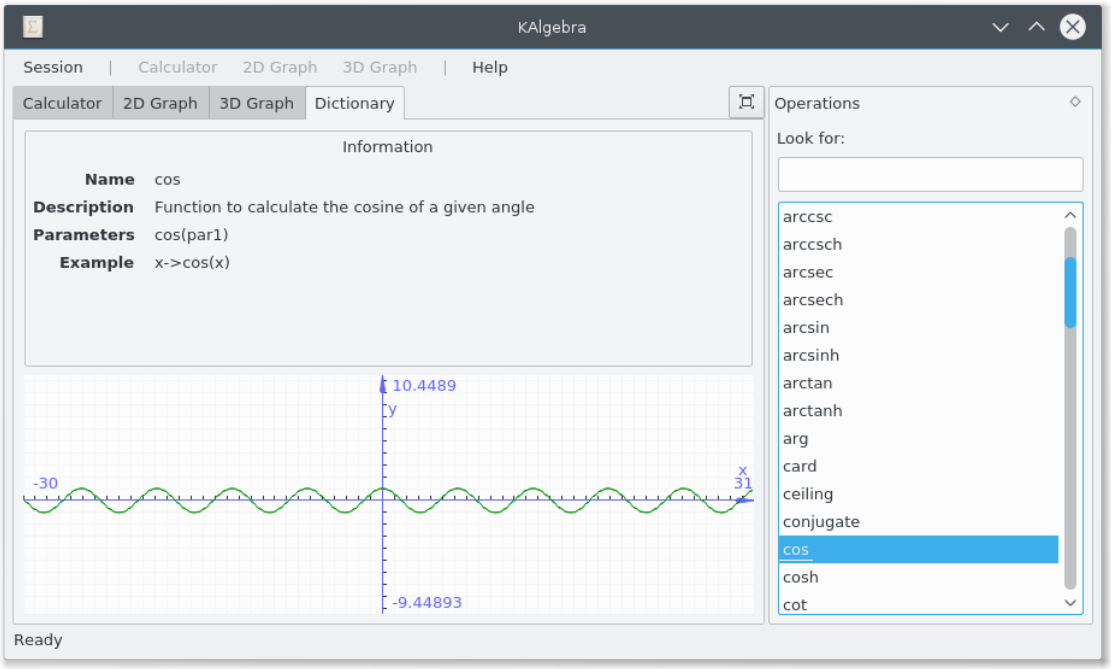

Session | Calculator 2D Graph 3D Graph | Help
Calculator 2D Graph 3D Graph Dictionary
(x,y)->5*sin((x^2+y^2)^.5)/(x^2+y^2)^.5
Ready

Capítulo 6

Diccionario

El diccionario proporciona una lista de todas las funciones integradas en KAlgebra. Se puede usar para encontrar la definición de una operación y sus parámetros de entrada. Es un lugar útil para encontrar las muchas funcionalidades que posee KAlgebra.

A continuación se muestra una captura de la ventana de la búsqueda de la función coseno en el diccionario de KAlgebra.



Capítulo 7

Órdenes permitidas por KAlgebra

7.1. más

- Nombre: más
- Descripción: adición
- Parámetros: más(... parámetros, ...)
- Ejemplo: x->x+2

7.2. veces

- Nombre: veces
- Descripción: multiplicación
- Parámetros: veces(... parámetros, ...)
- Ejemplo: x->x*2

7.3. menos

- Nombre: menos
- Descripción: Sustracción. Eliminará todos los valores desde el primero.
- Parámetros: menos(... parámetros, ...)
- Ejemplo: x->x-2

7.4. dividir

- Nombre: dividir
- Descripción: división
- Parámetros: dividir(par1, par2)
- Ejemplo: x->x/2

7.5. cociente

- Nombre: cociente
- Descripción: cociente
- Parámetros: cociente(par1, par2)
- Ejemplo: x->quotient(x, 2)

7.6. potencia

- Nombre: potencia
- Descripción: potenciación
- Parámetros: potencia(par1, par2)
- Ejemplo: x->x^2

7.7. radicación

- Nombre: radicación
- Descripción: radicación
- Parámetros: radicación(par1, par2)
- Ejemplo: x->root(x, 2)

7.8. factorial

- Nombre: factorial
- Descripción: factorial. factorial(n)=n!
- Parámetros: factorial(par1)
- Ejemplo: x->factorial(x)

7.9. y

- Nombre: y
- Descripción: y lógico
- Parámetros: y(... parámetros, ...)
- Ejemplo: x->piecewise { and(x>-2, x<2) ? 1, ? 0 }

7.10. o

- Nombre: o
- Descripción: o lógico
- Parámetros: o(... parámetros, ...)
- Ejemplo: x->piecewise { or(x>2, x>-2) ? 1, ? 0 }

7.11. xor

- Nombre: xor
- Descripción: xor lógico
- Parámetros: xor(... parámetros, ...)
- Ejemplo: x->piecewise { xor(x>0, x<3) ? 1, ? 0 }

7.12. negación

- Nombre: negación
- Descripción: negación lógica
- Parámetros: negación(par1)
- Ejemplo: x->piecewise { not(x>0) ? 1, ? 0 }

7.13. mcd

- Nombre: mcd
- Descripción: máximo común divisor
- Parámetros: mcd(... parámetros, ...)
- Ejemplo: x->mcd(x, 3)

7.14. mcm

- Nombre: mcm
- Descripción: mínimo común múltiplo
- Parámetros: mcm(... parámetros, ...)
- Ejemplo: x->mcm(x, 4)

7.15. resto

- Nombre: resto
- Descripción: resto
- Parámetros: resto(par1, par2)
- Ejemplo: x->resto(x, 5)

7.16. factorof

- Nombre: factorof
- Descripción: el factor de
- Parámetros: factorof(par1, par2)
- Ejemplo: x->factorof(x, 3)

7.17. max

- Nombre: max
- Descripción: máximo
- Parámetros: max(... parámetros, ...)
- Ejemplo: x->max(x, 4)

7.18. min

- Nombre: min
- Descripción: mínimo
- Parámetros: min(... parámetros, ...)
- Ejemplo: x->min(x, 4)

7.19. lt

- Nombre: lt
- Descripción: menor que. lt(a,b)=a<b
- Parámetros: lt(par1, par2)
- Ejemplo: x->piecewise { x<4 ? 1, ? 0 }

7.20. gt

- Nombre: gt
- Descripción: mayor que. gt(a,b)=a>b
- Parámetros: gt(par1, par2)
- Ejemplo: x->piecewise { x>4 ? 1, ? 0 }

7.21. eq

- Nombre: eq
- Descripción: igual. eq(a,b) = a=b
- Parámetros: eq(par1, par2)
- Ejemplo: x->piecewise { x=4 ? 1, ? 0 }

7.22. neq

- Nombre: neq
- Descripción: no igual. neq(a,b)=a≠b
- Parámetros: neq(par1, par2)
- Ejemplo: x->piecewise { x!=4 ? 1, ? 0 }

7.23. leq

- Nombre: leq
- Descripción: menor o igual. leq(a,b)=a≤b
- Parámetros: leq(par1, par2)
- Ejemplo: x->piecewise { x<=4 ? 1, ? 0 }

7.24. geq

- Nombre: geq
- Descripción: mayor o igual. geq(a,b)=a≥b
- Parámetros: geq(par1, par2)
- Ejemplo: x->piecewise { x>=4 ? 1, ? 0 }

7.25. implies

- Nombre: implies
- Descripción: implicación lógica
- Parámetros: implies(par1, par2)
- Ejemplo: x->piecewise { implies(x<0, x<3) ? 1, ? 0 }

7.26. approx

- Nombre: approx
- Descripción: aproximación. approx(a)=a±n
- Parámetros: approx(par1, par2)
- Ejemplo: x->piecewise { approx(x, 4) ? 1, ? 0 }

7.27. abs

- Nombre: abs
- Descripción: valor absoluto. abs(n)=|n|
- Parámetros: abs(par1)
- Ejemplo: x->abs(x)

7.28. floor

- Nombre: floor
- Descripción: función piso. floor(n)=⌊n⌋
- Parámetros: floor(par1)
- Ejemplo: x->floor(x)

7.29. ceiling

- Nombre: ceiling
- Descripción: función techo. ceil(n)=⌈n⌉
- Parámetros: ceiling(par1)
- Ejemplo: x->ceiling(x)

7.30. sin

- Nombre: sin
- Descripción: función para calcular el seno del ángulo indicado
- Parámetros: sin(par1)
- Ejemplo: x->sin(x)

7.31. cos

- Nombre: cos
- Descripción: función para calcular el coseno del ángulo indicado
- Parámetros: cos(par1)
- Ejemplo: x->cos(x)

7.32. tan

- Nombre: tan
- Descripción: función para calcular la tangente del ángulo indicado
- Parámetros: tan(par1)
- Ejemplo: x->tan(x)

7.33. sec

- Nombre: sec
- Descripción: secante
- Parámetros: sec(par1)
- Ejemplo: x->sec(x)

7.34. csc

- Nombre: csc
- Descripción: cosecante
- Parámetros: csc(par1)
- Ejemplo: x->csc(x)

7.35. cot

- Nombre: cot
- Descripción: cotangente
- Parámetros: cot(par1)
- Ejemplo: x->cot(x)

7.36. sinh

- Nombre: sinh
- Descripción: seno hiperbólico
- Parámetros: sinh(par1)
- Ejemplo: x->sinh(x)

7.37. cosh

- Nombre: cosh
- Descripción: coseno hiperbólico
- Parámetros: cosh(par1)
- Ejemplo: x->cosh(x)

7.38. tanh

- Nombre: tanh
- Descripción: tangente hiperbólica
- Parámetros: tanh(par1)
- Ejemplo: x->tanh(x)

7.39. sech

- Nombre: sech
- Descripción: secante hiperbólica
- Parámetros: sech(par1)
- Ejemplo: x->sech(x)

7.40. csch

- Nombre: csch
- Descripción: cosecante hiperbólica
- Parámetros: csch(par1)
- Ejemplo: x->csch(x)

7.41. coth

- Nombre: coth
- Descripción: cotangente hiperbólica
- Parámetros: coth(par1)
- Ejemplo: x->coth(x)

7.42. arcsin

- Nombre: arcsin
- Descripción: arcoseno
- Parámetros: arcsin(par1)
- Ejemplo: x->arcsin(x)

7.43. arccos

- Nombre: arccos
- Descripción: arcocoseno
- Parámetros: arccos(par1)
- Ejemplo: x->arccos(x)

7.44. arctan

- Nombre: arctan
- Descripción: arcotangente
- Parámetros: arctan(par1)
- Ejemplo: x->arctan(x)

7.45. arccot

- Nombre: arccot
- Descripción: arcocotangente
- Parámetros: arccot(par1)
- Ejemplo: x->arccot(x)

7.46. arccosh

- Nombre: arccosh
- Descripción: arcocoseno hiperbólico
- Parámetros: arccosh(par1)
- Ejemplo: x->arccosh(x)

7.47. arccsc

- Nombre: arccsc
- Descripción: arcocosecante
- Parámetros: arccsc(par1)
- Ejemplo: x->arccsc(x)

7.48. arccsch

- Nombre: arccsch
- Descripción: arcocosecante hiperbólica
- Parámetros: arccsch(par1)
- Ejemplo: x->arccsch(x)

7.49. arcsec

- Nombre: arcsec
- Descripción: arcosecante
- Parámetros: arcsec(par1)
- Ejemplo: x->arcsec(x)

7.50. arcsech

- Nombre: arcsech
- Descripción: arcosecante hiperbólica
- Parámetros: arcsech(par1)
- Ejemplo: x->arcsech(x)

7.51. arcsinh

- Nombre: arcsinh
- Descripción: arcoseno hiperbólico
- Parámetros: arcsinh(par1)
- Ejemplo: x->arcsinh(x)

7.52. arctanh

- Nombre: arctanh
- Descripción: arcotangente hiperbólica
- Parámetros: arctanh(par1)
- Ejemplo: x->arctanh(x)

7.53. exp

- Nombre: exp
- Descripción: exponente (e^x)
- Parámetros: exp(par1)
- Ejemplo: x->exp(x)

7.54. ln

- Nombre: ln
- Descripción: logaritmo con base e
- Parámetros: ln(par1)
- Ejemplo: x->ln(x)

7.55. log

- Nombre: log
- Descripción: logaritmo con base 10
- Parámetros: log(par1)
- Ejemplo: x->log(x)

7.56. conjugate

- Nombre: conjugate
- Descripción: Conjugar
- Parámetros: conjugate(par1)
- Ejemplo: x->conjugate(x*i)

7.57. arg

- Nombre: arg
- Descripción: Arg
- Parámetros: arg(par1)
- Ejemplo: x->arg(x*i)

7.58. real

- Name: real
- Description: Real
- Parámetros: real(par1)
- Ejemplo: x->real(x*i)

7.59. imaginary

- Nombre: imaginary
- Descripción: imaginario
- Parámetros: imaginary(par1)
- Ejemplo: x->imaginary(x*i)

7.60. sum

- Nombre: sum
- Descripción: suma
- Parámetros: sum(par1 : var=desde..hasta)
- Ejemplo: x->x*sum(t*t:t=0..3)

7.61. product

- Nombre: product
- Descripción: producto
- Parámetros: product(par1 : var=desde..hasta)
- Ejemplo: x->product(t+t:t=1..3)

7.62. diff

- Nombre: diff
- Descripción: diferenciación
- Parámetros: diff(par1 : var)
- Ejemplo: x->(diff(x^2:x))(x)

7.63. card

- Nombre: card
- Descripción: cardinal
- Parámetros: card(par1)
- Ejemplo: x->card(vector { x, 1, 2 })

7.64. scalarproduct

- Nombre: scalarproduct
- Descripción: producto escalar
- Parámetros: scalarproduct(... parámetros, ...)
- Ejemplo: x->scalarproduct(vector { 0, x }, vector { x, 0 })[1]

7.65. selector

- Nombre: selector
- Descripción: selección del elemento de orden par1 de la lista o vector par2
- Parámetros: selector(par1, par2)
- Ejemplo: x->scalarproduct(vector { 0, x }, vector { x, 0 })[1]

7.66. union

- Nombre: union
- Descripción: une varios elementos del mismo tipo
- Parámetros: union(... parámetros, ...)
- Ejemplo: x->union(list { 1, 2, 3 }, list { 4, 5, 6 })[rem(floor(x), 5)+3]

7.67. forall

- Nombre: forall
- Descripción: para todo
- Parámetros: forall(par1 : var)
- Ejemplo: x->piecewise { forall(t:t@list { true, false, false }) ? 1, ? 0 }

7.68. exists

- Nombre: exists
- Descripción: existe
- Parámetros: exists(par1 : var)
- Ejemplo: x->piecewise { exists(t:t@list { true, false, false }) ? 1, ? 0 }

7.69. map

- Nombre: map
- Descripción: aplica una función a cada elemento de una lista
- Parámetros: map(par1, par2)
- Ejemplo: x->map(x->x+x, list { 1, 2, 3, 4, 5, 6 })[rem(floor(x), 5)+3]

7.70. filter

- Nombre: filter
- Descripción: elimina todos los elementos que no cumplen una condición
- Parámetros: filter(par1, par2)
- Ejemplo: x->filter(u->rem(u, 2)=0, list { 2, 4, 3, 4, 8, 6 })[rem(floor(x), 5)+3]

7.71. transponer

- Nombre: transponer
- Descripción: transponer
- Parámetros: transponer(par1)
- Ejemplo: x->transpose(matrix { matrixrow { 1, 2, 3, 4, 5, 6 } })[rem(floor(x), 5)+3][1]

Capítulo 8

Créditos y licencia

- Derechos de autor del programa 2005-2009 Aleix Pol.

Derechos de autor de la documentación 2007 Aleix Pol aleixpol@gmail.com.

Traducido por Leticia Martín Hernández leticia.martin@gmail.com y Eloy Cuadra ecuadra@eloihr.net.

Esta documentación está sujeta a los términos de la Licencia de Documentación Libre GNU.

Este programa está sujeto a los términos de la Licencia Pública General GNU.