

The KmPlot Handbook

Klaus-Dieter Möller and Philip Rodrigues



The KmPlot Handbook

Contents

1	Introduction	1
2	First Steps With KmPlot	3
2.1	Simple Function Plot	3
2.2	Edit Properties	3
3	Using KmPlot	5
3.1	Function Types	6
3.1.1	Explicit Functions	6
3.1.2	Parametric Functions	7
3.1.3	Entering Functions in Polar Coordinates	8
3.2	Combining Functions	8
3.3	Changing the appearance of functions	8
3.4	Popup menu	9
4	Configuring KmPlot	10
4.1	General Configuration	10
4.2	Colors Configuration	12
4.3	Coordinate System Configuration	12
4.3.1	The Axes Configuration	12
4.3.2	The Grid Configuration	13
4.4	Scaling Configuration	14
4.5	Fonts Configuration	15

The KmPlot Handbook

5 KmPlot Reference	16
5.1 Function Syntax	16
5.2 Predefined Function Names and Constants	16
5.3 Extensions	17
5.4 Mathematical Syntax	18
5.5 Plotting Area	18
5.6 Cross Hair Cursor	19
6 Command Reference	20
6.1 The File Menu	20
6.2 The Edit Menu	20
6.3 The Plot Menu	21
6.4 The Zoom Menu	21
6.5 The Tools Menu	22
6.6 The Settings Menu	22
6.7 The Help Menu	22
7 Scripting KmPlot	24
8 Developer's Guide to KmPlot	29
9 Credits and License	30
A Installation	31

Abstract

KmPlot is a mathematical function plotter for the KDE Desktop.

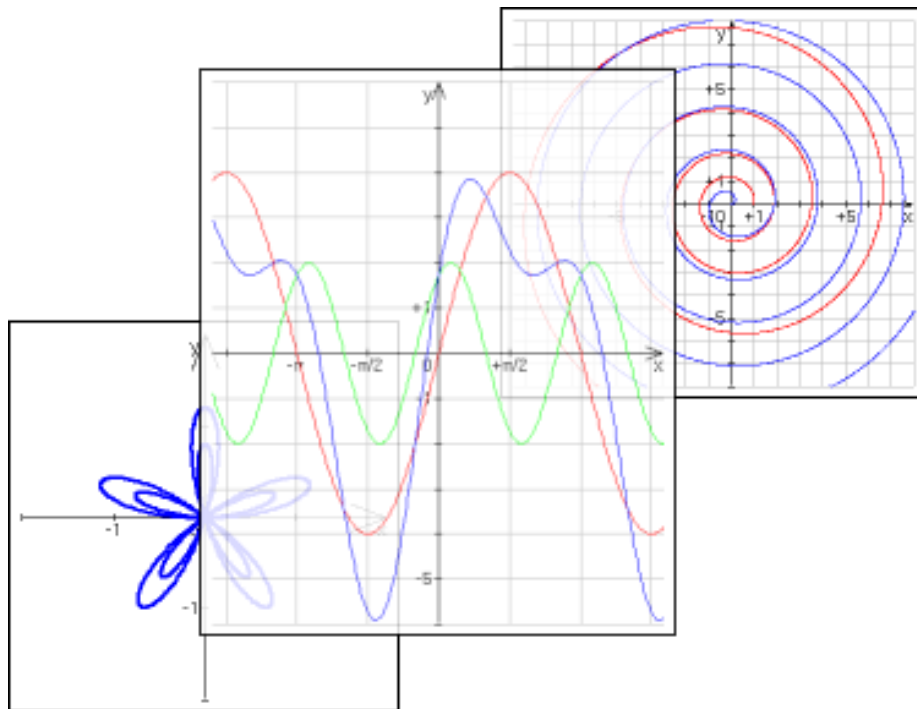


KmPlot is part of the KDE-EDU Project: <http://edu.kde.org/>

Chapter 1

Introduction

KmPlot is a mathematical function plotter for the KDE Desktop. It has a powerful built-in parser. You can plot different functions simultaneously and combine them to build new functions.



KmPlot supports parametric functions and functions in polar coordinates. Several grid modes are supported. Plots may be printed with high precision in the correct scale.

KmPlot also provides some numerical and visual features like:

The KmPlot Handbook

- Filling and calculating the area between the plot and the first axis
- Finding maximum and minimum values
- Changing function parameters dynamically
- Plotting derivatives and integral functions.

These features help in learning the relationship between mathematical functions and their graphical representation in a coordinate system.

Chapter 2

First Steps With KmPlot

2.1 Simple Function Plot

In the main toolbar there is a simple text box in which you can enter a function expression. Simply enter:

```
x^2
```

and press **Enter**. This will draw the plot of $y=x^2$ in the coordinate system. Enter another expression in the text box like

```
5*sin(x)
```

and another plot will be added.

Click on one of the lines you have just plotted. Now the cross hair gets the color of the plot and is attached to the plot. You can use the mouse to move the cross hair along the plot. In the status bar at the bottom of the window the coordinates of the current position is displayed. Note that if the plot touches the x-axis the root will be displayed in the status bar, too.

Click the mouse again and the cross hair will be detached from the plot.

2.2 Edit Properties

Let us make some changes to the function and change the color of the plot.

You can edit all functions with the Plot → Edit Plots... menu entry. A dialog appears which lists all the functions that you have plotted. Notice that KmPlot has automatically found a unique function name for your expressions and completed the expression to a function equation.

The KmPlot Handbook

Select $f(x)=x^2$ in the list. A double click or pressing the Edit button will show you a dialog window. Here you have access to a lot of options. Let us rename the function and move the plot 5 units down. Change the function equation to

```
parabola(x)=x^2-5
```

To select another color for the plot click into the Color: box. Finally press OK and your changes take effect in the coordinate system.

NOTE

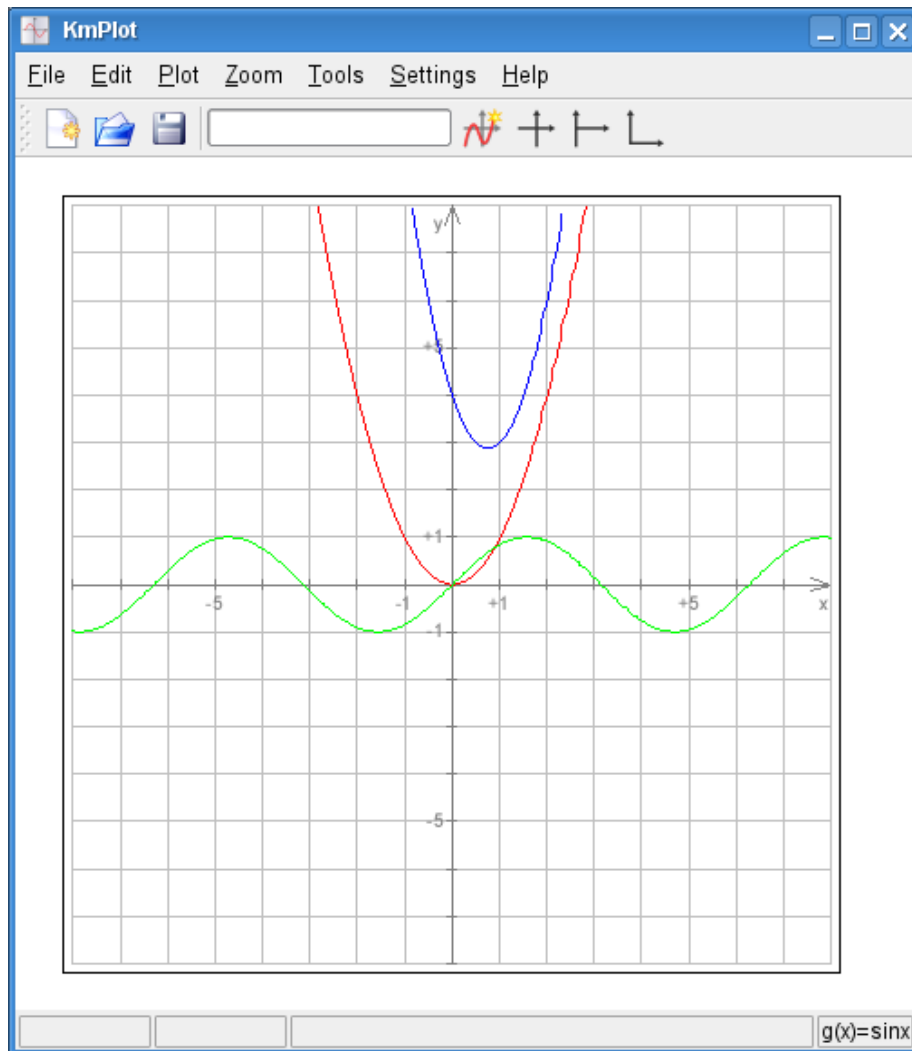
All changes can be undone until you press OK in the Edit Plots dialog.

Chapter 3

Using KmPlot

KmPlot deals with named functions, which can be specified in terms of Cartesian coordinates (called 'explicit functions'), polar coordinates or as parametric functions. To enter a function, choose Plot → Edit Plots.... You can also enter new functions in the Function equation text box in the main KmPlot window. The text box can handle explicit and polar functions. Each function you enter must have a unique name (i.e., a name that is not taken by any of the existing functions displayed in the list box). A function name will be automatically generated if you do not specify one.

For more information on KmPlot functions, see [chapter 5](#).



3.1 Function Types

3.1.1 Explicit Functions

To enter an explicit function (i.e., a function in the form $y=f(x)$) into KmPlot, just enter it in the following form:

```
f(x)=expression
```

Where:

- f is the name of the function, and can be any string of letters and numbers you like, provided it does not start with any of the letters x , y or r (since these are used for parametric and polar functions).
- x is the x -coordinate, to be used in the expression following the equals sign. It is in fact a dummy variable, so you can use any variable name you like, but the effect will be the same.
- *expression* is the expression to be plotted, given in appropriate syntax for KmPlot. See Section 5.4.

As an example, to draw the graph of $y=x^2+2x$, enter the following into the functions dialog of KmPlot:

```
f ( x ) = x ^ 2 + 2 x
```

3.1.2 Parametric Functions

Parametric functions are those in which the x and y coordinates are defined by separate functions of another variable, often called t . To enter a parametric function in KmPlot, follow the procedure as for an explicit function, but prefix the name of the function describing the x -coordinate with the letter x , and the function describing the y -coordinate with the letter y . As with explicit functions, you may use any variable name you wish for the parameter. To draw a parametric function, you must go to PlotNew Parametric Plot.... A function name will be created automatic if you do not specify one.

As an example, suppose you want to draw a circle, which has parametric equations $x=\sin(t)$, $y=\cos(t)$. In the KmPlot functions dialog, do the following:

1. Open the parametric plot dialog with Plot \rightarrow New Parametric Plot....
2. Enter a name for the function, say, `circle`, in the Name box. The names of the x and y functions change to match this name: the x function becomes `xcircle(t)` and the y function becomes `ycircle(t)`.
3. In the x and y boxes, enter the appropriate equations, i.e., `xcircle(t)=sin(t)` and `ycircle(t)=cos(t)`.

Click on OK and the function will be drawn.

You can set some further options for the plot in this dialog:

Hide If this option is selected, the plot is not drawn, but KmPlot remembers the function definition, so you can use it to define other functions.

Custom plot minimum-range, Custom plot maximum-range If this options are selected, you can change the maximum and minimum values of the parameter t for which the function is plotted using the Min: and Max: boxes.

Line width: With this option you can set the width of the line drawn on the plot area, in units of 0.1mm.

Color: Click on the color box and pick a color in the dialog that appears. The line on the plot will be drawn in this color.

3.1.3 Entering Functions in Polar Coordinates

Polar coordinates represent a point by its distance from the origin (usually called r), and the angle a line from the origin to the point makes with the x -axis (usually represented by the Greek letter θ). To enter functions in polar coordinates, use the menu entry Plot \rightarrow New Polar Plot.... In the box labeled r , complete the function definition, including the name of the θ variable you want to use, e.g., to draw the Archimedes' spiral $r=\theta$, enter:

```
r(theta)=theta
```

so that the whole line reads 'r(theta)=theta'. Note that you can use any name for the θ variable, so 'r(foo)=foo' would have produced exactly the same output.

3.2 Combining Functions

Functions can be combined to produce new ones. Simply enter the functions after the equals sign in an expression as if the functions were variables. For example, if you have defined functions $f(x)$ and $g(x)$, you can plot the sum of f and g with:

```
sum(x)=f(x)+g(x)
```

Note that you can only combine functions of the same type, e.g. an explicit function cannot be combined with a polar function.

3.3 Changing the appearance of functions

To change the appearance of a function's graph on the main plot window, select the function in the Edit Plots dialog, and click on the Edit button. In the dialog which appears, you can change the line width in the text box, and the color of the function's graph by clicking on the color button at the bottom. If you are editing an explicit function, you will see a dialog with three tabs. In the first one you specify the equation of the function. The Derivatives tab lets you draw the first and second derivative to the function. With the Integral tab you can draw the integral of the function which is calculated using Euler's method.

Another way to edit a function is to right click on the graph. In the popup menu that appears, choose Edit

For more information on the popup menu, see Section 3.4.

3.4 Popup menu

When right-clicking on a plot function or a single-point parametric plot function a popup menu will appear. In the menu there are five items available:

Hide Hides the selected graph. Other plots of the graph's function will still be shown.

Remove Removes the function. All its graphs will disappear.

Edit Shows the editor dialog for the selected function.

Copy Copies the graph to another running KmPlot instance.

Move Moves the graph to another running KmPlot instance.

For plot functions the following four items are also available:

Get y-Value Opens a dialog in which you can find the y-value corresponding to a specific x-value. The selected graph will be highlighted in the dialog. Enter an x value in the X: box, and click on Calculate (or press **Enter**). The corresponding y value will be shown under Y:.

Search for Minimum Value Find the minimum value of the graph in a specified range. The selected graph will be highlighted in the dialog that appears. Enter the lower and upper boundaries of the region in which you want to search for a minimum, and click Find. The x and y values at the minimum will be shown.

Search for Maximum Value This is the same as Search for Minimum Value above, but finds maximum values instead of minima.

Calculate Integral Select the x-values for the graph in the new dialog that appears. Calculates the integral and draws the area between the graph and the x-axis in the selected range in the color of the graph.

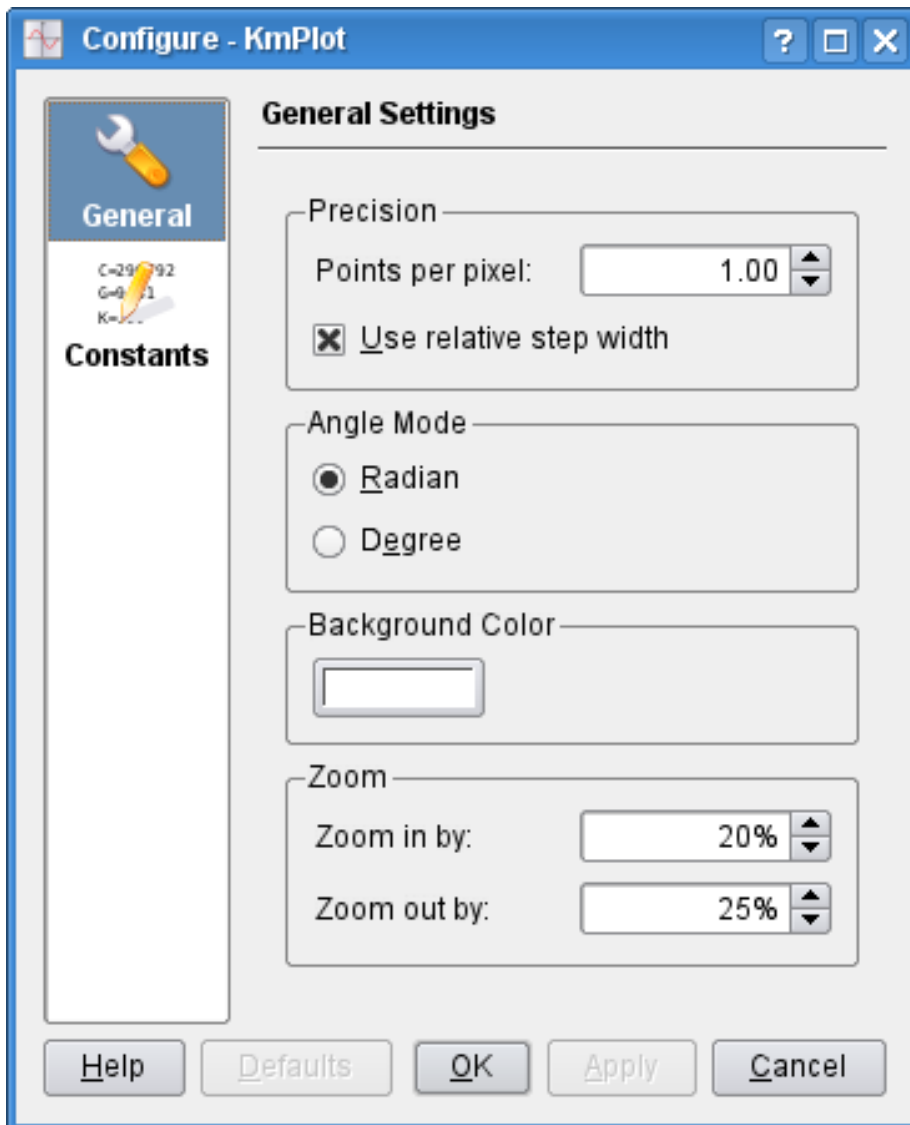
Chapter 4

Configuring KmPlot

To access the KmPlot configuration dialog, select Settings → Configure KmPlot.... A number of settings (Colors..., Coordinate System..., Scaling... and Fonts...) can only be changed from the Edit menu.

4.1 General Configuration

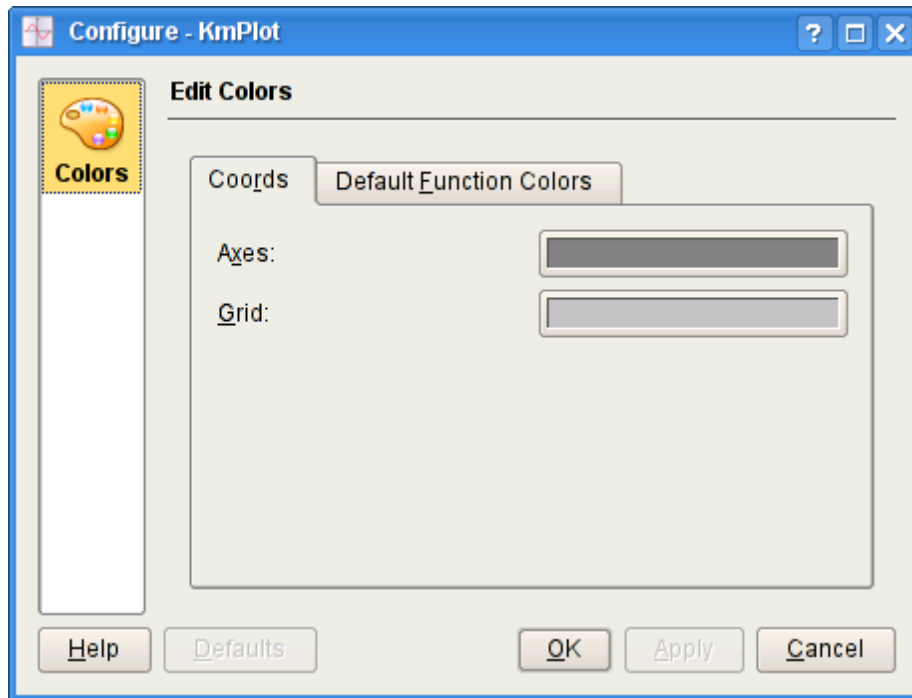
Here you can set global settings which automatic will be saved when you exit KmPlot. In the first page you can set calculation-precision, angle-mode (radians and degrees), background color and zoom in and zoom out factors.



The second page let you define you own constants. KmPlot saves the constants in the same file as KCalc does. That means you can create a constant in KmPlot, close the program and load it in KCalc and vice versa. KmPlot only supports constant names that consist of one capital character and if you in KCalc define a constant name that is not one character, the name will be truncated. E.g, if you already have the constants "apple" and "bananas" in KCalc, they will be renamed to "A" and "B" in KmPlot.

4.2 Colors Configuration

In the Coords tab of the Colors configuration dialog, you can change the colors of the axes and grid of the main KmPlot area.



In the Default Function Colors tab, you can change the colors used for the graphs of the ten functions allowed in KmPlot.

4.3 Coordinate System Configuration

4.3.1 The Axes Configuration

X-Axis Sets the range for the x-axis scale. You can choose one of the predefined ranges, or select Custom to make your own. Note that in the Custom boxes, you can use the predefined functions and constants (see Section 5.2) as the extremes of the range (e.g., set Min: to 2π). You can even use functions you have defined to set the extremes of the axis range. For example, if you have defined a function $f(x) = x^2$, you could set Min: to $f(3)$, which would make the lower end of the range equal to 9.

Y-Axis Sets the range for the y-axis. See 'X-Axis' above.

Axis-line width: Sets the width of the lines representing the axes.

- Tic width:** Sets the width of the lines representing tics on the axes.
- Tic length:** Sets the length of the lines representing tics on the axes.
- Show labels** If checked, the names (x, y) of the axes are shown on the plot and the axes' tics are labeled.
- Show extra frame** If checked, the plot area is framed by an extra line.
- Show axes** If checked, the axes are visible.
- Show arrows** If checked, the axes are displayed with arrows at their ends.

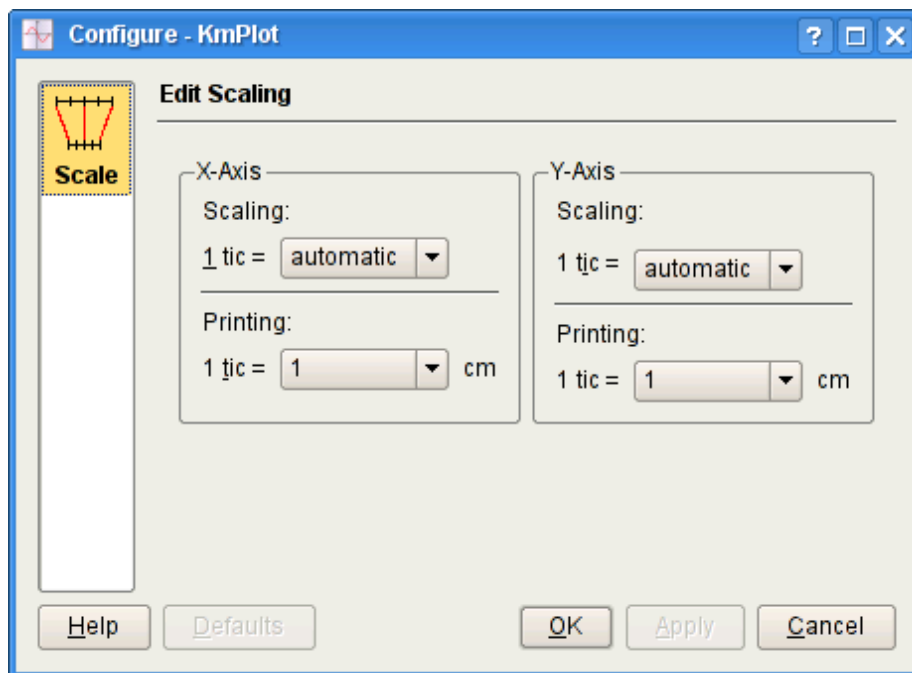
4.3.2 The Grid Configuration

You can set the Grid Style to one of four options:

- None** No gridlines are drawn on the plot area
- Lines** Straight lines form a grid of squares on the plot area.
- Crosses** Crosses are drawn to indicate points where x and y have integer values (e.g., $(1,1)$, $(4,2)$ etc.).
- Polar** Lines of constant radius and of constant angle are drawn on the plot area.

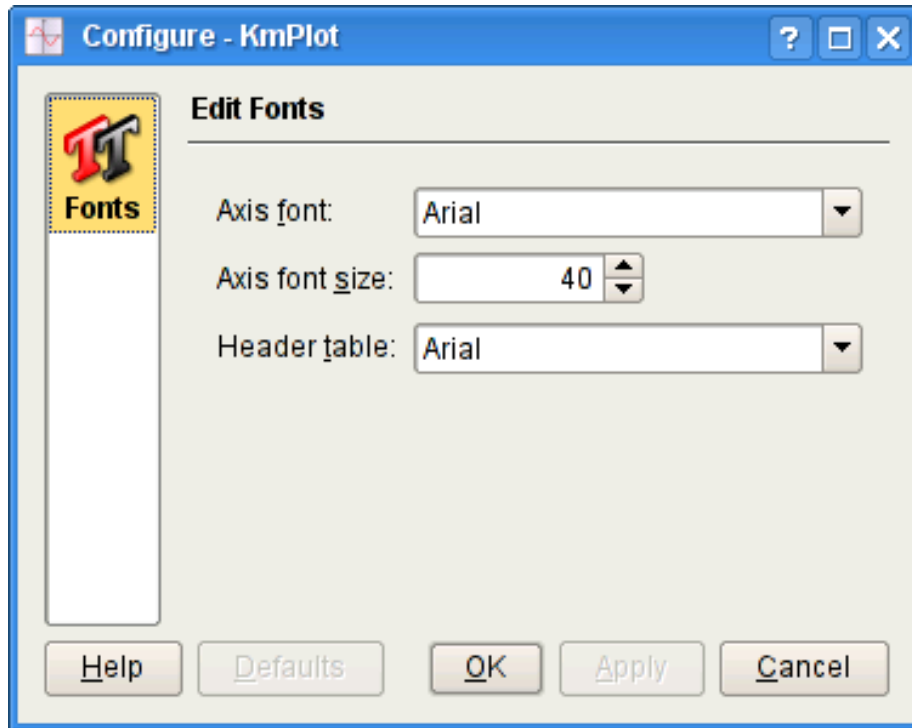
The Line width option is used to set the width of the lines of the grid.

4.4 Scaling Configuration



For each axis, you can set the **Scaling:** and **Printing:** of one tic. The **Scaling:** option selects how many units apart the axis tics will be (and therefore, how far apart grid lines will be drawn), and the **Printing:** option selects the length of one tic when displayed on the screen or printed. In this way, these options can be used to change the size of the graph on screen or on a page: For example, doubling the **Printing:** setting whilst keeping the **Scaling:** setting the same will result in the graph doubling in in height or width.

4.5 Fonts Configuration



Header table: sets the font for the information table shown in KmPlot print-outs, and Axis font: and Axis font size: sets the font and its size used for all labels on the axes in the plot area.

Chapter 5

KmPlot Reference

5.1 Function Syntax

Some syntax rules must be complied with:

```
name(var1[, var2])=term [;extensions]
```

name The function name. If the first character is 'r' the parser assumes that you are using polar coordinates. If the first character is 'x' (for instance 'xfunc') the parser expects a second function with a leading 'y' (here 'yfunc') to define the function in parametric form.

var1 The function's variable

var2 The function 'group parameter'. It must be separated from the function's variable by a comma. You can use the group parameter to, for example, plot a number of graphs from one function. The parameter values can be selected manually or you can choose to have a slider bar that controls one parameter. By changing the value of the slider the value parameter will be changed. The slider can be set to an integer between 0 and 100.

term The expression defining the function.

5.2 Predefined Function Names and Constants

All the predefined functions and constants that KmPlot knows can be shown by selecting Help → Predefined Math Functions. They are:

sqr, sqrt Return the square and square root of a number, respectively.

exp, ln Return the exponential and natural logarithm of a number, respectively.

log Returns the logarithm to base 10 of a number.

sin, arcsin Return the sine and inverse sine of a number, respectively. Note that the argument to sin and the return value of arcsin are in radians.

cos, arccos Return the cosine and inverse cosine of a number, respectively. Also in radians.

tan, arctan Return the tangent and inverse tangent of a number, respectively. Also in radians.

sinh, arcsinh Return the hyperbolic sine and inverse hyperbolic sine of a number, respectively.

cosh, arccosh Return the hyperbolic cosine and inverse hyperbolic cosine of a number, respectively.

tanh, arctanh Return the hyperbolic tangent and inverse hyperbolic tangent of a number, respectively.

sin, arcsin Return the sine and inverse sine of a number, respectively. Note that the argument to sin and the return value of arcsin are in radians.

cos, arccos Return the cosine and inverse cosine of a number, respectively. Also in radians.

pi, e Constants representing π (3.14159...) and e (2.71828...), respectively.

These functions and constants and even all user defined functions can be used to determine the axes settings as well. See Section 4.3.1.

5.3 Extensions

An extension for a function is specified by entering a semicolon, followed by the extension, after the function definition. The extension can either be written in the Quick Edit box or by using the DCOP method Parser addFunction. None of the extensions are available for parametric functions but N and D[a,b] work for polar functions too. For example:

```
f(x)=x^2; A1
```

will show the graph $y=x^2$ with its first derivative. Supported extensions are described below:

N The function will be stored but not be drawn. It can be used like any other user-defined or predefined function.

- A1 The graph of the derivative of the function will be drawn additionally with the same color but less line width.
- A2 The graph of the second derivative of the function will be drawn additionally with the same color but less line width.
- D[a,b] Sets the domain for which the function will be displayed.
- P[a,b,...] Give a set of values of a group parameter for which the function should be displayed. For example: $f(x, k) = k * x; P[1, 2, 3]$ will plot the functions $f(x) = x$, $f(x) = 2 * x$ and $f(x) = 3 * x$. You can also use functions as the arguments to the P option.

Please note that you can do all of these operations by using the function editor dialog too.

5.4 Mathematical Syntax

KmPlot uses a common way of expressing mathematical functions, so you should have no trouble working it out. The operators KmPlot understands are, in order of decreasing precedence:

- ^ The caret symbol performs exponentiation. e.g., 2^4 returns 16.
- *, / The asterisk and slash symbols perform multiplication and division . e.g., $3 * 4 / 2$ returns 6.
- +, - The plus and minus symbols perform addition and subtraction. e.g., $1 + 3 - 2$ returns 2.

Note the precedence, which means that if parentheses are not used, exponentiation is performed before multiplication/division, which is performed before addition/subtraction. So $1 + 2 * 4^2$ returns 33, and not, say 144. To override this, use parentheses. To use the above example, $((1 + 2) * 4)^2$ will return 144.

5.5 Plotting Area

By default, explicitly given functions are plotted for the whole of the visible part of the x-axis. You can specify an other range in the edit-dialog for the function. For every pixel on the x-axis KmPlot calculates a function value. If the plotting area contains the resulting point it is connected to the last drawn point by a line.

Parametric functions are plotted for parameter values from 0 up to 2 π . You can set the plotting range in the dialog for the function too.

5.6 Cross Hair Cursor

While the mouse cursor is over the plotting area the cursor changes to a cross hair. The current coordinates can be seen at the intersections with the coordinate axes and also in the status bar at the bottom of the main window.

You can trace a function's values more precisely by clicking onto or next to a graph. The selected function is shown in the status bar in the right column. The cross hair then will be caught and be colored in the same color as the graph. If the graph has the same color as the background color, the cross hair will have the inverted color of the background. When moving the mouse or pressing the keys Left or Right the cross hair will follow the function and you see the current x- and y-value. If the cross hair is close to y-axis, the root-value is shown in the statusbar. You can switch function with the Up and Down keys. A second click anywhere in the window or pressing any non-navigating key will leave this trace mode.

Note that tracing is only possible with explicitly given functions. The coordinates are always displayed according to a Cartesian system of coordinates. Neither non-single-point parametric functions nor functions given in polar coordinates can be traced in this way.

Chapter 6

Command Reference

6.1 The File Menu

File → **New (Ctrl+N)** Starts a new Plot by clearing the coordinate system and resetting the function parser.

File → **Open... (Ctrl+O)** Opens an existing document.

File → **Open Recent** Displays a list of recently opened files. Selecting one from this list plots the functions in the file.

File → **Save (Ctrl+S)** Saves the document.

File → **Save As...** Saves the document under another name.

File → **Print... (Ctrl+P)** Sends the plot to a printer or file.

File → **Export...** Export values to a textfile. Every value in the parameter list will be written to one line in the file.

File → **Quit (Ctrl+Q)** Exits KmPlot.

6.2 The Edit Menu

Edit → **Colors...** Displays the Colors Settings dialog box. See Section 4.2.

Edit → **Coordinate System...** Displays the Coordinate System dialog box. See Section 4.3.

Edit → **Scaling...** Displays the Scale Settings dialog box. See Section 4.4.

Edit → **Fonts...** Displays the Fonts Settings dialog box. See Section 4.5.

Edit → **Coordinate System I** Show both positive and negative x- and y-values on the grid.

Edit → **Coordinate System II** Show positive and negative y-values, but positive x-values only

Edit → **Coordinate System III** Show only positive x- and y-values.

6.3 The Plot Menu

Plot → **New Function Plot...** Opens the dialog for creating a new function plot. See chapter 3.

Plot → **New Parametric Plot...** Opens the dialog for creating a new parametric plot. See chapter 3.

Plot → **New Polar Plot...** Opens the dialog for creating a new polar plot. See chapter 3.

Plot → **Edit Plots...** Displays the functions dialog. There you can add, edit and remove functions. See chapter 3.

6.4 The Zoom Menu

The first five items in the menu change zoom-mode.

Zoom → **No Zoom (Ctrl+0)** Disable the zoom-mode.

Zoom → **Zoom Rectangular (Ctrl+1)** Let the user draw a rectangle. The minimum and maximum values will be set to the coordinates of the rectangle.

Zoom → **Zoom In (Ctrl+2)** The minimum and maximum values will come closer to each other and the selected point in the graph will be centered.

Zoom → **Zoom Out (Ctrl+3)** The minimum and maximum values will be more separated from each other and the selected point in the graph will be centered.

Zoom → **Center Point (Ctrl+4)** The selected point in the graph will be centered.

Zoom → **Fit Widget to Trigonometric Functions** The scale will be adapted to trigonometric functions. This works both for radians and degrees.

6.5 The Tools Menu

This menu contains some tools for the functions that can be useful:

Tools → **Get y-Value...** Let the user get the y-value from a specific x-value. At the moment, only plot functions are supported. Type a value or expression in the text box under "X:". In the list below all the available functions are shown. Press the "Calculate" button to find the function's y-value. The result will be shown in the y-value box.

Tools → **Search for Minimum Value...** Find the minimum value of the graph in a specified range.

Tools → **Search for Maximum Value...** Find the maximum value of the graph in a specified range.

Tools → **Calculate Integral** Select a graph and the x-values in the new dialog that appears. Calculates the integral and draws the area between the graph and the x-axis in the range of the selected x-values in the color of the graph.

6.6 The Settings Menu

Settings → **Show/Hide Toolbar** Toggle on and off the display of the toolbar. The default is on.

Settings → **Show/Hide Statusbar** Toggle on and off the display of the status bar at the bottom of the KmPlot main window. The default is on.

Settings → **Full Screen Mode (Ctrl-Shift-F)** With this action you toggle the full screen mode.

Settings → **Show Sliders** Toogles the display of sliders 1 to 4 on and off.

Settings → **Configure Shortcuts...** Personalize the keybindings for KmPlot.

Settings → **Configure Toolbars...** Personalize the toolbars for KmPlot.

Settings → **Configure KmPlot...** Customize KmPlot. The options available to you are described in chapter 4.

6.7 The Help Menu

KmPlot has a standard KDE Help as described below, with one addition:

Help → **Predefined Math Functions...** Opens a window with a list of the predefined function names and constants that KmPlot knows.

The KmPlot Handbook

The standard KDE Help entries are:

Help → **KmPlot Handbook (F1)** Invokes the KDE Help system starting at the KmPlot help pages. (this document).

Help → **What's This? (Shift+F1)** Changes the mouse cursor to a combination arrow and question mark. Clicking on items within KmPlot will open a help window (if one exists for the particular item) explaining the item's function.

Help → **Report Bug...** Opens the Bug report dialog where you can report a bug or request a 'wishlist' feature.

Help → **About KmPlot** This will display version and author information.

Help → **About KDE** This displays the KDE version and other basic information.

Chapter 7

Scripting KmPlot

A new feature in KDE 3.4 is that you can write scrips for KmPlot with DCOP. For example, if you want to define a new function $f(x)=2\sin x+3\cos x$, set its line width to 20 and then draw it, you type in a console:

dcop kmplot-PID Parser addFunction "f(x)=2sin x+3cos x" As a result, the new function's id number will be returned, or -1 if the function could not be defined.

>dcop kmplot-PID Parser setFunctionFLineWidth 20 ID This command sets the function with the id number ID the line width to 20.

>dcop kmplot-PID View drawPlot This command repaints the window so that the function get visible.

A list over the available functions:

KmPlotShell fileOpen &url Load the file *url*.

MainDlg isModified Returns true if any changes are done.

MainDlg editColors Opens the color edit dialog.

MainDlg editAxes Opens the coordinate system edit dialog.

MainDlg editScaling Opens the scaling edit dialog.

MainDlg editFonts Opens the fonts edit dialog.

MainDlg editConstants Opens the constants edit dialog.

MainDlg newFunction Opens the new function plot dialog.

MainDlg newParametric Opens the new parametric plot dialog.

MainDlg newPolar Opens the new polar plot dialog.

MainDlg toggleShowSlider0 Shows/hides parameter slider window number 1.

The KmPlot Handbook

- MainDlg toggleShowSlider1** Shows/hides parameter slider window number 2.
- MainDlg toggleShowSlider2** Shows/hides parameter slider window number 3.
- MainDlg toggleShowSlider3** Shows/hides parameter slider window number 4.
- MainDlg slotSave** Saves the functions (opens the save dialog if it is a new file).
- MainDlg slotSaveas** The same as choosing File → Save As in the menu.
- MainDlg slotEditPlots** Opens the edit plots dialog.
- MainDlg slotPrint** Opens the print dialog.
- MainDlg slotExport** Opens the export dialog.
- MainDlg slotSettings** Opens the settings dialog.
- MainDlg slotNames** Shows a list of predefined math functions.
- MainDlg slotCoord1** Coordinate System I.
- MainDlg slotCoord2** Coordinate System II.
- MainDlg slotCoord3** Coordinate System III.
- MainDlg getYValue** The same as choosing Tools → Get y-Value... in the menu.
- MainDlg findMinimumValue** The same as choosing Tools → Search for Minimum Value... in the menu.
- MainDlg findMaximumValue** The same as choosing Tools → Search for Maximum Value... in the menu.
- MainDlg graphArea** The same as choosing Tools → Calculate Integral in the menu.
- Parser addFunction f_str** Adds a new function with the expression f_str . If the expression does not contain a function name, it will be auto-generated. The id number of the new function is returned, or -1 if the function couldn't be defined.
- Parser delfkt id** Removes the function with the id number id . If the function could not be deleted, false is returned, otherwise true.
- Parser setFunctionExpression f_str id** Sets the expression for the function with the id number id to f_str . Returns true if it succeed, otherwise false.
- Parser countFunctions** Returns the number of functions (parametric functions are calculated as two).
- Parser listFunctionNames** Returns a list with all functions.
- Parser fnameToId f_str** Returns the id number of f_str or -1 if the function name f_str was not found.

The KmPlot Handbook

- Parser id x** Calculates the value x for the function with the ID id or returns 0.0 if id does not exist.
- Parser functionFVisible id** Returns true if the function with the ID id is visible, otherwise false.
- Parser functionF1Visible id** Returns true if the first derivative of the function with the ID id is visible, otherwise false.
- Parser functionF2Visible id** Returns true if the second derivative of the function with the ID id is visible, otherwise false.
- Parser functionIntVisible id** Returns true if the integral of the function with the ID id is visible, otherwise false.
- Parser setFunctionFVisible visible id** Shows the function with the ID id if $visible$ is true. If $visible$ is false, the function will be hidden. True is returned if the function exists, otherwise false
- Parser setFunctionF1Visible visible id** Shows the first derivative of the function with the ID id if $visible$ is true. If $visible$ is false, the function will be hidden. True is returned if the function exists, otherwise false.
- Parser setFunctionF2Visible visible id** Shows the second derivative of the function with the ID id if $visible$ is true. If $visible$ is false, the function will be hidden. True is returned if the function exists, otherwise false.
- Parser setFunctionIntVisible visible id** Shows the integral of the function with the ID id if $visible$ is true. If $visible$ is false, the function will be hidden. True is returned if the function exists, otherwise false.
- Parser functionStr id** Returns the function expression of the function with the ID id . If the function not exists, an empty string is returned instead.
- Parser functionFColor id** Returns the color of the function with the ID id .
- Parser functionF1Color id** Returns the color of the first derivative of the function with the ID id .
- Parser functionF2Color id** Returns the color of the second derivative of the function with the ID id .
- Parser functionIntColor id** Returns the color of the integral of the function with the ID id .
- Parser setFunctionFColor color id** Sets the color of the function with the ID id to $color$. True is returned if the function exists, otherwise false.
- Parser setFunctionF1Color color id** Sets the color of the first derivative of the function with the ID id to $color$. True is returned if the function exists, otherwise false.
- Parser setFunctionF2Color color id** Sets the color of the second derivative of the function with the ID id to $color$. True is returned if the function exists, otherwise false.
- Parser setFunctionIntColor color id** Sets the color of the integral of the function with the ID id to $color$. True is returned if the function exists, otherwise false.
- Parser functionFLineWidth id** Returns the line width of the function with the ID id . If the function not exists, 0 is returned.

- Parser functionF1LineWidth id** Returns the line width of the first derivative of the function with the ID *id*. If the function not exists, 0 is returned.
- Parser functionF2LineWidth id** Returns the line width of the first derivative of the function with the ID *id*. If the function not exists, 0 is returned.
- Parser functionIntLineWidth id** Returns the line width of the integral of the function with the ID *id*. If the function not exists, 0 is returned.
- Parser setFunctionFLineWidth linewidth id** Sets the line width of the function with the ID *id* to *linewidth*. True is returned if the function exists, otherwise false.
- Parser setFunctionF1LineWidth linewidth id** Sets the line width of the first derivative of the function with the ID *id* to *linewidth*. True is returned if the function exists, otherwise false.
- Parser setFunctionF2LineWidth linewidth id** Sets the line width of the second derivative of the function with the ID *id* to *linewidth*. True is returned if the function exists, otherwise false.
- Parser setFunctionIntLineWidth linewidth id** Sets the line width of the integral of the function with the ID *id* to *linewidth*. True is returned if the function exists, otherwise false.
- Parser functionParameterList id** Returns a list with all the parameter values for the function with the ID *id*.
- Parser functionAddParameter new_parameter id** Adds the parameter value *new_parameter* to the function with the ID *id*. True is returned if the operation succeed, otherwise false.
- Parser functionRemoveParameter remove_parameter id** Removes the parameter value *remove_parameter* from the function with the ID *id*. True is returned if the operation succeed, otherwise false.
- Parser functionMinValue id** Returns the minimum plot range value of the function with the ID *id*. If the function not exists or if the minimum value is not defined, an empty string is returned.
- Parser functionMaxValue id** Returns the maximum plot range value of the function with the ID *id*. If the function not exists or if the maximum value is not defined, an empty string is returned.
- Parser setFunctionMinValue min id** Sets the minimum plot range value of the function with the ID *id* to *min*. True is returned if the function exists and the expression is valid, otherwise false.
- Parser setFunctionMaxValue max id** Sets the maximum plot range value of the function with the ID *id* to *max*. True is returned if the function exists and the expression is valid, otherwise false.
- Parser functionStartXValue id** Returns the initial x point for the integral of the function with the ID *id*. If the function not exists or if the x-point-expression is not defined, an empty string is returned.
- Parser functionStartYValue id** Returns the initial y point for the integral of the function with the ID *id*. If the function not exists or if the y-point-expression is not defined, an empty string is returned.

The KmPlot Handbook

Parser setFunctionStartXValue min id Sets the initial x point for the integral of the function with the ID id to x . True is returned if the function exists and the expression is valid, otherwise false.

Parser setFunctionStartYValue max id Sets the initial y point for the integral of the function with the ID id to y . True is returned if the function exists and the expression is valid, otherwise false.

View stopDrawing If KmPlot currently is drawing a function, the procedure will stop.

View drawPlot Redraws all functions.

Chapter 8

Developer's Guide to KmPlot

If you want to contribute to KmPlot feel free to send a mail to kd.moeller@t-online.de or f_edemar@linux.se

Chapter 9

Credits and License

KmPlot

Program copyright 2000-2002 Klaus-Dieter Möller kd.moeller@t-online.de

CONTRIBUTORS

- CVS: Robert Gogolok mail@robert-gogoloh.de
- Porting GUI to KDE 3 and Translating: Matthias Messmer bmlmessmer@web.de
- Various improvements: Fredrik Edemar f_edemar@linux.se

Documentation copyright 2000--2002 by Klaus-Dieter Möller kd.moeller@t-online.de.

Documentation extended and updated for KDE 3.2 by Philip Rodrigues phil@kde.org.

Documentation extended and updated for KDE 3.3 by Philip Rodrigues phil@kde.org
and Fredrik Edemar f_edemar@linux.se.

Documentation extended and updated for KDE 3.4 by Fredrik Edemar f_edemar@linux.se.

This documentation is licensed under the terms of the [GNU Free Documentation License](#).

This program is licensed under the terms of the [GNU General Public License](#).

Appendix A

Installation

KmPlot is part of the KDE project <http://www.kde.org/> .

KmPlot can be found in the kdedu package on <ftp://ftp.kde.org/pub/kde/> , the main FTP site of the KDE project.



KmPlot is part of the KDE EDU Project: <http://edu.kde.org/>

KmPlot has its own homepage on [SourceForge](#). You can also find archives of older versions of KmPlot there, for example, for KDE 2.x

In order to compile and install KmPlot on your system, type the following in the base directory of the KmPlot distribution:

```
% ./configure
% make
% make install
```

Since KmPlot uses **autoconf** and **automake** you should have no trouble compiling it. Should you run into problems please report them to the KDE mailing lists.