Graph2D

Kai Nickel

COLLABORATORS			
	TITLE : Graph2D		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Kai Nickel	June 8, 2025	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Graph2D

1.1 Table of contents

G r a p h 2 D

Functionplotter and -analyzer
Version 3.10 (26.09.1995)
Author: Kai Nickel

3. Appendix

1.2 Short description

1.1 Short description

Graph2D is a function-plotter that can create graphs of math. functions and is able to discuss them. The graphs can be two- or threedimensional systems or SIRDS (Magic3D).

You can derive and simplify the functions, calculate value-tables, integrate functions numerically, create tangents and taylor-approximations, etc.

The Graph2D package contains an installer-script, online-help and is localized to German and English.

Feature overview:

- Plots functions into highly configurable 2D or 3D coordinate-systems
- Generates SIRDS for three-dimensional functions
- Work with many functions in many windows at the same time
- Generation of discussions (zero-points, extrema, turning-points, monotony, symmetry)
- symbolic derivation and numerical integration of funktion-terms
- taylor-approximation
- use your own functions to define other functions
- Generation of value-tables and tangents
- Shows functions in "mathematical notation"
- system- and user-friendly MUI-interface
- Printing of graphs and discussions
- Save graphs or notation as IFF pictures
- Online-Help as AmigaGuide
- Installation with the Commodore-Installer
- uncrippeled shareware

System requirements:

- Amiga with Kickstart2.0 or higher
- installed MUI-System version 2.3 or higher

1.3 Author

1.2 Author

Graph2D and the documentation were written by

Kai Nickel

Mail : Herzogstrasse 29
67435 Neustadt
Germany

eMail : un7x@rz.uni-karlsruhe.de
kai@rpsbbs.rlp.org

I would be glad about bug reports or ideas concerning future development of Graph2D.

About this documentation:

As you can see my English is not very good, and I know that there are a lot of mistakes in this documentation. Especially the mathematical expressions are often pure "inventions" of me because I could not find them in my dictionaries. But despite this I hope that you can at least figure out what I wanted to say... If anybody wants to put this documentation or the catalog into "real" English or even translate it into any other language:
please contact me!

How to get updates:

I will send updates of Graph2D to Aminet/misc/math. But you may also call the german Graph2D-Support-BBS (24h online) called RPSBBS:

Modem: 49-6329-1624 (HST/V32b V42b)
ISDN : 49-6329-990015 (X75)
Login: graph2d
Path : /Lokal/Support/Amiga/Graph2D
(my username is "Kai")

1.4 Copyright

1.3 Copyright

with Graph2D on it must not be higher than US$3. The inclusion into FD-series is hereby permitted, when nothing of the above said things is injured. You must not change any of the files in the original archive or remove or add files from or to it. It must remain complete and unchanged!

The author cannot guarantee the correct function of Graph2D and cannot be made responsible for any negative consequences that may result from the use of Graph2D. Updates or bugfixes are not guaranteed.

Graph2D was developed with the Amiga-Oberon-Compiler V3.20 by the A+L AG. The license of that compiler prohibits the use of Graph2D in the military sector.

Graph2D uses the MUI-system by Stefan Stuntz.

MagicWB and some of the icons in the Graph2D package are copyright by Martin Huttenloher.

The SIRDS algorithm was developed by Kilian Singer and has been included to Graph2D with his permission.

1.5 Registration

1.4 Registration

Graph2D is shareware. Everybody using the program someway serious, is asked to register himself at the author.

Anybody using Graph2D longer than a certain evaluation period (lets say 2 weeks) without registering breaks the licence. The unregistered version is not limited in any way but there are nerv-requesters that appear sometimes and remind you about registering. Please concern that I have invested many month of work in Graph2D and I just want to have a kind of positive feedback for my efforts. This may also motivate me for further developments or bugfixes.

After registering you get a personal keyfile which gives you the right to use Graph2D and future updates (if there are any) permanently. Registration of course also makes the nerv requesters disappear!

How to register:

You send me a letter containing the money and a filled out Graph2D.RegForm because I need your address to generate your personal keyfile.

If you want to get your keyfile via eMail the registration fee is US$15 or DEM20.
If you want me to send you the keyfile on a disc in a letter the registration fee us US$20 or DEM25 because sending a disc around the world is quite expensive.

Just include the money in cash to the letter. And please don’t use other currencies than US$ or DEM.

If anything went wrong and you did not get a response from me after lets say 4 weeks you should consider contacting me with another mail (don't get me wrong: You shall not send the money twice :-))

Thank you for supporting the shareware idea!

1.6 Installation

1.5 Installation

Normal:

The best way to get Graph2D installed is to start the script "Graph2D-Install". You can start Graph2D after the installation just by clicking it from the workbench.

Don't be afraid: If you chose "expert user" at the installation you may confirm really every action!

Manually:

If you don't have the Commodore installer you alternatively can also install Graph2D by hand:

- Create a Graph2D drawer and copy the main program into it
- Copy the guide and the reg-form of your preferred language into the drawer
- Copy the "GarbageCollector.library" to the "LIBS:" drawer of your system or to PROGDIR:Libs/.
- Copy the "Graph2D.catalog" to your system's "Locale:Catalogs/English" or to PROGDIR:Catalogs/Libs/
- optional: Copy the subdirectories "Functions" and "Graphs" to your drawer
- optional: Replace the icons by the icons of the "MWBIcons" drawer

Originalarchive:

The original Graph2D-archive contains the following files. If files have been added or removed then you have an illegal copy!

Graph2D	- main program
Graph2D-Install	- installation script
Docs/English/Graph2D.guide	- documentation/onlinehelp
Docs/English/Graph2DHistory.guide	- development history
Docs/English/Graph2D.RegForm	- registration form
Docs/English/Graph2D.ReadMe	- description
Docs/English/Product-Info	- description in fish-format

```
  Docs/Deutsch/Graph2D.guide          - german documentation/onlinehelp
  Docs/Deutsch/Graph2DHistory.guide   - german history
  Docs/Deutsch/Graph2D.RegForm        - german registration form
  Docs/Deutsch/Graph2D.LiesMich       - german description
  Catalogs/english/Graph2D.catalog    - english catalog
  Libs/garbagecollector.library       - library
  Libs/GarbageCollector.LiesMich      - german infotext for the library
  Graphs/Standard.2D                  - 2D-standard coordinatesystem
  Graphs/Standard.3D                  - 3D-standard coordinatesystem
  Graphs/Standard.SIRDS               - SIRDS-standard coordinatesystem
  Graphs/Trigonometrie.2D             - example of a trigonometrical system
  Funktions/Bsp2D.fkt                 - example of a functionlist
  Funktions/Bsp3D.fkt                 - example of a list of 3D-functions
  MWBIcons/ClickForColors             - MagicWB-palette (try it!)
  MWBIcons/Graph2D.info               - MagicWB-style icon
  MWBIcons/Graph2D.guide.info         - MagicWB-style icon
  MWBIcons/Graph2DHistory.guide.info - MagicWB-style icon
  MWBIcons/Graph2D.RegFrom.info       - MagicWB-style icon
  MWBIcons/Drawer.info                - MagicWB-style icon
  MWBIcons/Funktions.info             - MagicWB-style icon
  MWBIcons/Graphs.info                - MagicWB-style icon
  MWBIcons/Catalogs.info              - MagicWB-style icon
  MWBIcons/Libs.info                  - MagicWB-style icon
```

1.7 Q&A

1.6 Questions and answers

So how exactly can I display the graph of a function?

First you have to generate a new function via "New", enter the functionterm (e.g. "f(x)=sin(x)^2"), press <return> to interpret the function and open a "New Graph". To make the function be plotted into the graph-window you now have to mark the function as "plot" and "Redraw" the graph to make the cange visible. The function should now be visible - of course only, if parts of the function are inside the choosen interval of the graph. You can change the visible interval, if you "edit" the graph.

1.8 Basics

2.1 Basics

Graph2D should be easy to use and look nice with every font and configuration thanks to MUI.

This documentation is also available as context-sensitive online help. You can read the help just by pressing the "Help"-key on the keyboard or by selecting "Help..." from the "Project"-menu.

Some windows of Graph2D contain a status-line that shows up short

help-texts for the objects the mouse-pointer is actually placed on.

After the start by double-clicking the Graph2D icon a main-window - the so called functioneditor - opens in which you can enter and edit functions.

If there are functions present, you can look at them in graph-windows or mathematically analyze them.

1.9 Project-menu

2.1.1 The project-menu

The project-menu is available from the function-editor, from graph- and notation-windows.

Project

Functioneditor...	- Open the function editor
New Graph...	- Creates a new graph-window. The graph type (2D-graph or SIRDS-graph) has to be specified in a little requester.
Presets...	- Opens the presets-editor
Help...	- Opens the context-sensitive online-help like pressing the HELP-key
About...	- Opens an info-window
Iconify	- Closes all windows and shows an icon on the workbench to reopen them again
Quit	- Quit Graph2D

1.10 Function editor

2.2 Function editor

The funktioneditor is a window, in that all present functions are listed and can be edited.

A function from the list-gadget appears in the function gadget by clicking on the function. If finish the entered or modified string with pressing <return>, the functionterm gets automatically interpreted.

If the interpreter detects an error in the function, the error gets displayed in the "status line" when your mouse pointer is placed over the function gadget. Additionally the "bad" function will appear in italics in the function-list. Of course such a bad function cannot be plotted or analyzed!

Pressing the HELP-key over the function or error gadget shows immediately the "Functions"-chapter of the documentation where you can find information about the function-syntax.

In the string-gadget "Graph" with the popup-list all existing graphs are shown by their names. You can give an individual name to every graph. When you double-click a graph in the list it will appear in the foreground.

A double-click to a function toggles the visible-mark as if you had clicked onto the visible-gadget. A function that appears in the selected graph ist marked with a preceding ">" in the function-list. Changes been made on a visible function will then be executed when you click Redraw in the graph-window - not before! This allows you to do more than one change at once without a time-consuming redraw after every change. You can change this behavior if you activate the auto-redraw mode in the presets-editor.

The image-buttons stand for the following things you can do with the selected function:

New Graph...	- Creates a new graph-window. The graph type (2D-graph or SIRDS-graph) has to be specified in a little requester.
Presets...	- Opens the presets-editor
Simplify	- A simplified aequivalent of the selected function will be added to the list
Derive	- The derivation of the selected function will be added to the list
Diskussion	- Initiates a selectable discussion
num. integration	- Calculates the integral
Value-table	- Creates a value-table
Tangente	- Creates a tangente
math. notation	- Opens the notation-window with the selected function
Rexx	- Execute an ARexx command

As long as the function-editor is active, the following menus can be selected:

Project - The project-menu

Functions

Delete all	- The functionlist will be deleted
Load ...	- Load a functionlist
Append...	- Append a functionlist (without deleting the actual one)
Save as...	- Save the functionliste

Edit

New	- A new, "empty" function will be added to the list at the selected position and can be immediately edited

```
  Cut                 - The selected function will be cut out of the list
  Paste               - A former cut-out function will be added to the
                        list at the selected position

Miscellanious

  Simplify            - like the according image-button
  Derive              - "
  Diskussion          - "
  num. integration... - "
  Value-table...      - "
  Tangente ...        - "
  math. notation...   - "

Rexx                  - ARexx
```

1.11 math. notation

```
2.2.1 Mathematical notation
```

In a window the selected function will be displayed in a more readable, a more "normal" way. Divisions are shown by horizontal lines and exponents are really at high-position. I call this the "mathematical notation"...

If the notation exceeds the visible range of the window you can scroll around with the scrollbars. The window gets closed by pressing the close-gadget. You can have opened more than one notation-windows at the same time an work in another part of Graph2D without problems.

You can use the following menus:

```
Project      - The project-menu

Notation

  Print...            - Prints the notation via Workbench-drivers and
                        settings. You can edit a scaling factor here.
  Save IFF...         - Save the notation as IFF picture.
```

1.12 Discussion

```
2.2.2 Discussion
```

Graph2D can discuss fuction with one variable in a selectable test-interval. This interval should always be choosen as small as possible for best results. Graph2D shows a report of the results in a textviewer and may be printed from there.

A dicussion may take (especially on 68000er Amigas) sometimes quite a long time and it is also very recommended to verify the results by having a look to the graph of the function. So you probably will see soon, that

certain types of zero-points will sometimes really not be detected.

A complete discussion consists of the following parts:

Zero-points: f(x) = 0

The zero-points of the selected function will be detected. In addition to the x-value you will also get the following information about the type of the zero-point:

```
-+       f'(x) > 0
+-       f'(x) < 0
++       f'(x) = 0
--       f'(x) = 0
```

Extrema: f'(x) = 0

```
min      f''(x) > 0
max      f''(x) < 0
```

Turning-points: f''(x) = 0

```
left->right   f'''(x) < 0
right->left   f'''(x) > 0
```

Monotony:

The test-intervall will be divided into parts where the selected function is:

```
monotonous raising       :  f'(x) >= 0
monotonous falling       :  f'(x) <= 0
strong monotonous raising:  f'(x) >  0
strong monotonous falling:  f'(x) <  0
```

Symmetry:

In this part the selected function will be tested whether it is

- symmetric to the origin: f(x) = - f(-x)
- symmetric to the y-axis: f(x) =   f(-x)

If both is not correct, then "No symmetry" will be reported.

1.13 num. integration

2.2.3 numerical integration

Allows you to calculate the integral of the selected function in an intervall you have to specify via the "Simpson"-formula.

```
a, b: Integrationlimits
k   : Subdivisions
```

```
  a
  /            b-a
  | f(x)dx = ----- * ( f(x ) + 4*f(x ) + 2*f(x ) + ...
  /            k*3         0         1         2
  b

              ... + 2*f(x   ) + 4*f(x   ) + f(x ) )
                          k-2         k-1       k
```

1.14 Value-table

2.2.4 Value-table

A value-table of the selected function in a definable intervall and with a selectable step-width will be generated.

1.15 Tangente

2.2.5 Tangente

A new function will be added to the list which represents a tangete on a definable point x of the selected function.

1.16 Taylor

2.2.6 Taylor

A function can be approximated in a range around a selectable point x0 with the Taylor-polynom. The higher you choose the degree of that polynom the more exact the approximation will be. But beware of choosing "degree" too high (> 10) because the need of memory and time will increase dramatically! The calculated polynom will be inserted into your function list and is marked with a preceding "T" in its name.

Taylor:

```
   oo       (n)
  ----     f   (x ) *  x
  \              0      0              n
   |     ------------------- * (x - x )
  /                                   0
  ----           n!
  n=0
```

Graph2D cannot sum up infinite terms and uses the following expression to

generate a Tylor-polynome with a selectable degree n:

$$\sum_{k=0}^{n} \left(\sum_{l=k}^{n} \frac{f^{(k)}(x_0) * (-x_0)^l}{l! * (k-l)!} \right) * x^k$$

1.17 Graph windows

2.3 Graph windows

Graph-windows can be opened by the user at any number. There are three types of graphs:

2D, 3D, SIRDS

Which function gets displayed or not in a graph-window can be selected in the functioneditor. The graph-windows and the function-editor run parallel so that you can switch between them and work simultaneously in both of them. Graph-windows are resizeable.

1.18 2D-graph-window

2.3.1 2D-graph-window

2D-graph-windows can be opened by the user at any number. With the graphs you can look at functions (also more than one in a graph) in a coordinate-system.

Which function gets displayed or not in a graph-window can be selected in the functioneditor. The graph-windows and the function-editor run parallel so that you can switch between them and work simultaneously in both of them. Graph-windows are resizeable.
The present intervall and the name of the graph get displayed in the graph-windows title. As long as you press the left mousebutton the coordinates according to the mouseposition get displayed in the text-line above the coordinate-system.
You can scroll around in the graph with the cursor keys.

The image-buttons have the following meanings:

New Graph - Creates a new graph-window. The graph type (2D-graph or SIRDS-graph) has to be specified in a little requester.
Presets - Opens the presets-editor

```
  Redraw    - The graph gets recalculated showing all changes that have
              been made to the functions since the last time
  Edit      - Open the 2D-graph-editor
  Zoom in   - After marking an area with the mouse a new graph
              opens displaying just the selected area
  Zoom out  - A new graph opens containing a four-times larger
              visible intervall

  Print     - Prints the visible graph
  SaveIFF   - Saves graph as IFF-picture. The pixel-size
              can be choosen in the presets-editor


  As long as a 2D-graph-window is active, you can select one of the
following menus:


Project               - The project-menu

Graph

  Redraw              - like the according image-button
  Zoom in             -             "
  Zoom out            -             "
  Print               -             "
  SaveIFF...          - Saves graph as IFF-picture. The pixel-size
                        can be choosen in the preset-editor.
  Close               - Close the graph window like pressing on the
                        regular close-gadget

Settings

  Axis...             - open the 2D-graph-editor
                        with active page "Axis"
  Functions...        - open the 2D-graph-editor
                        with active page "Functions"
  Misc...             - open the 2D-graph-editor
                        with active page "Misc"
  Window...           - open the 2D-graph-editor
                        with active page "Window"
  Save...             - save the graph settings
  Load...             - load graph settings
```

1.19 2D-graph-editor

```
2.3.2 2D-graph-editor


  With this editor you are able to adjust the visible intervall, the
scaling of the axis and some other attributes of a graph. This editor is
divided into four pages:


Page: Axis
```

You can set the following attributes for each the x- and the y-axis:

from ... to - The visible intervall for the axis. The value also depends of the choosen unit!

Unit - The basical unit of the axis. The other values of the axis will be (internally) multiplied with with the unit - so the unit is a kind of a scaling factor. Usually you set the unit to "1". But maybe if you want to generate a coordinate-system especially for trigonometric functions a unit of "3.141" would make sense. If you then enter a visible intervall of e.g. [-2; 4] it would mean, that the axis in fact shows an intervall of [-2*3.141; 4*3.141] = [-6.282; 12.564]. You can chose the unit free as long it stays >0.

Draw axis - Display the axis or not. Even if displayed not, the visible intervall of an axis is of course of importance!

Title - You can give a title to every axis which will be shown in the graph next to the axis. If you maybe chose a unit of e.g. "3.141" then it is important, to make that "public" by e.g. titeling the axis as "x-axis in pi" or something like this.

Marks - An axis can optionally be divided by many small marks. Now here you can define after how many units a mark should be drawn. Furthermore you may place a certain number of (smaller) "Submarks" between every two regular marks. A submark-value of "0" means: no submarks. If you choose too many marks for a graph so that one could not visually tell one from another then Graph2D will not draw the marks.

Text - The marks can optionally be equipped with numbers next to them. You can decide after how many marks a number should be drawn. The number again is dependant of the unit you have chosen. If you draw too many numbers so that they would overleap one another, Graph2D just leaves out some of them.

Page: Functions

For every function drawn to this graph can be set individual display-parameters. The list entry "< default settings >" has got a special meaning: the settings made for this entry are used for the next functions you will draw into the graph. The default settings are also saved and loaded together with the other graph-settings.

"Color" is just the color the function will have.

"Accuracy" is the precision the function will be calculated in. At a lower accuracy only fewer function-values will be calculated and the output-speed will increase - but the graph of the function may get a bit unprecise ans show some "steps".

"Pen" makes the axis be drawn "bold" or "normal". The "bold" mode takes more time but can be useful for high resolution prints.

"Connect" makes Graph2D to connect the single calculated function-values by lines. So if the graph of a function "jumps" in the visible intervall, maybe because of a low accuracy, the otherwise existing gaps can be

avoided. Of the other hand "connect" could draw connections that in reality just do not exist (e.g. sgn(x))!

"Pattern" lets you choose the pattern of the lines that "connect" the dots of the function.

Page: Misc

Now you can decide for both axis whether there should be a "Raster", what kind of raster ("type") it will be and how large the distance between the rasterlines in x- and y-direction should be. Again: this is unit-dependant.

This "Pen" here makes the axis be drawn "bold" or "normal". The "bold" mode takes more time but can be useful for high resolution prints.

"Font" allows you to choose the font for the whole graph.

Page: Window

Toggle the "image buttons" and "status-line" on or off for the graph window.

You can load and save Graph-settings with any filename with "Load..." and "Save...".

After leaving the graph-editor with "Ok" the active graph gets redrawn to make the probably made changes visible. When leaving the editor via "Cancel" or simply closing the window nothing will be changed of course and the old settings stay valid.

1.20 3D-graph-window

2.3.3 3D-graph-window

3D-graph-windows can be opened by the user at any number. With the graphs you can look at functions in a three-dimensional coordinate-system.

Which function gets displayed or not in a graph-window can be selected in the functioneditor. The graph-windows and the function-editor run parallel so that you can switch between them and work simultaneously in both of them. Graph-windows are resizeable. The present intervall and the name of the graph get displayed in the graph-windows title.

The two sliders on the right and bottom of the window allow you to adjust the eye-perspective. Try it with a low accuracy first because the graph gets updated after every change.

The image buttons:

New Graph - Creates a new graph-window. The graph type

```
              (2D-graph or SIRDS-graph) has to be specified
              in a little requester.
  Presets   - Opens the presets-editor


  Redraw    - will cause the graph to become actualized and to
              make all changes visible that may have been made to the
              function since the last redraw.
  Edit      - open the 3D-graph-editor

  Print     - Prints the visible graph
  SaveIFF   - Saves graph as IFF-picture. The pixel-size
              can be choosen in the presets-editor
```

As long as a 3D-graph-window is active, you can select one of the following menus:

```
Project        - The project-menu

Graph

  Redraw       - like the according image button
  Print...     - "
  SaveIFF...   - Saves graph as IFF-picture. The pixel-size
                 can be choosen in the preset-editor.
  Close        - Close the graph window like pressing on the
                 regular close-gadget

Settings

  Axis...      - open the 3D-graph-editor with
                 active page "Axis"
  Colors...    - open the 3D-graph-editor with
                 active page "Colors"
  Misc...      - open the 3D-graph-editor with
                 active page "Misc"
  Window...    - open the 3D-graph-editor with
                 active page "Window"
  Save...      - save the graph settings
  Load...      - load graph settings
```

1.21 3D-graph-editor

```
2.3.4 3D-graph-editor
```

With this editor you are able to adjust the visible intervall, the scaling of the axis and some other attributes of a 3D-graph. This editor is divided into four pages:

```
Page: Axis

You can set the following attributes for each the x, y and z-axis:
```

from ... to - The visible intervall for the three axis.

Page: Color

The 3D-graph of a function looks like a "patchwork" constisting of areas with two different colors. "Function1" and "Function2" are those colors.
The colors of the axis and the area-borders inside the function can be adjusted with "Axis"

Page: Misc

"Accuracy" is the precision with that a graph should be calculated. At a lower accuracy only fewer function-values will be calculated and the output-speed will increase - but the graph of the function may get a bit unprecise, show some "steps" and the areas become larger.

With "Axis" you can choose whether the coordinate-system should be drawn as a box, as ordinary axis or at at all.

Page: Window

Toggle the "image buttons" and "status-line" on or off for the graph window.

You can load and save Graph-settings with any filename with "Load..." and "Save...".

After leaving the graph-editor with "Ok" the active graph gets redrawn to make the probably made changes visible. When leaving the editor via "Cancel" or simply closing the window nothing will be changed of course and the old settings stay valid.

1.22 SIRDS-graph-window

2.3.5 SIRDS-graph-window

SIRDS-graph-windows can be opened by the user at any number. With the graphs you can look at functions in a three-dimensional SIRDS coordinate-system.

Which function gets displayed or not in a graph-window can be selected in the functioneditor. The graph-windows and the function-editor run parallel so that you can switch between them and work simultaneously in both of them. Graph-windows are resizeable. The present intervall and the name of the graph get displayed in the graph-windows title.

The image buttons:

New Graph - Creates a new graph-window. The graph type (2D-graph or SIRDS-graph) has to be specified in a little requester.
Presets - Opens the presets-editor

```
  Redraw    - will cause the graph to become actualized and to
              make all changes visible that may have been made to the
              function since the last redraw.
  Edit      - open the SIRDS-graph-editor

  Print     - Prints the visible graph
  SaveIFF   - Saves graph as IFF-picture. The pixel-size
              can be choosen in the presets-editor


  As long as a SIRDS-graph-window is active, you can select one of the
following menus:

Project         - The project-menu

Graph

  Redraw        - like the according image button
  Print...      - "
  SaveIFF...    - Saves graph as IFF-picture. The pixel-size
                  can be choosen in the preset-editor.
  Close         - Close the graph window like pressing on the
                  regular close-gadget

Settings

  Axis...       - open the SIRDS-graph-editor with active
                  page "Axis"
  Misc...       - open the SIRDS-graph-editor with active
                  page "Misc"
  Window...     - open the SIRDS-graph-editor with active
                  page "Window"
  Save...       - save the graph settings
  Load...       - load graph settings
```

1.23 SIRDS-graph-editor

```
2.3.6 SIRDS-graph-editor


  In this editor the visible intervall, the scaling of the axis and some
other attributes of a SIRDS-graph can be set. There are three pages:

Page: Axis

  For each the x-, y- and z-axis a visible intervall must be choosen x- and
y-axis run as known horizontally and vertically on the monitor while the
z-axis seems to "touch" the user magically...


Page: Misc

  With "colors" you can choose how many colors should be used to create the
SIRDS-graph. The number of colors has no effect on the output speed.
```

The "transparence" determines how "white" the graph should be. The higher the value is the more pixels are drawn in the background color (white). This looks better to some people and is very useful when you print a SIRDS in high density.

And last but not least you can choose an "Accuracy", with that a graph should be calculated. At a lower accuracy only fewer function-values will be calculated and the output-speed will increase - but the graph of the function may get a bit unprecise and show some "steps". But in a SIRDS graph the low resolutions are of a high importance because it sometimes may take really long to create a SIRDS...

Page: Window

Toggle the "image buttons" and "status-line" on or off for the graph window.

You can load and save graph-settings with any filename with "Load..." and "Save...".

After leaving the graph-editor with "Ok" the active graph gets redrawn to make the probably made changes visible. When leaving the editor via "Cancel" or simply closing the window nothing will be changed of course and the old settings stay valid.

1.24 Textviewer

2.4 Textviewer

Textviewers show all kind of texts and lists in Graph2D. With the scroller- bar you can scroll through the text, print it with "Print", save the text as ASCII with "Save..." and close the window with "Ok" or the close-gadget.

A textviewer does not have to been closed to work on with Graph2D!

1.25 Preset-editor

2.5 Presets-editor

You can do the following presets for Graph2D:

Settings directory - this path is used for the file-requesters in the graph editors

2D-graph - this 2D-settings file is used whenever a new 2D-graph will be opened

3D-graph - this 3D-settings file is used whenever a new 3D-graph will be opened

SIRDS-graph - this SIRDS-settings file is used whenever a new SIRDS-graph will be opened

IFF-size - the width and height in pixel a graph will have when saved as IFF

Auto-redraw - when active a graph will be refreshed immediately after a change been made to a drawn function. This includes adding or removing of a function.

Font - the font used for math. notation windows.

Print-Font - the font used for printing or saving a math. notation

Smartbuttons - if you don't like the image buttons in the functioneditor you can replace them by normal text buttons or nothing at all

"Use" uses the presets until the program ends, "Save" makes your selections permanent.

1.26 Print

2.6 Print-Requester

With this requester you can print a graphic via the workbench-printer-drivers and the workbench settings. The size of the print can be selected with width and "height".

1.27 ARexx

2.7 Arexx-port

The function editor contains a "Rexx"-menu that allows you to execute any ARexx script you want. Some of the scripts included in the Graph2D-package are predefined as own menu items, any other scripts can be selected with a file-requester.

Meaning of some of the included scripts:

SendToMathScript - Send the function that is selected in the function-editor to the program MathScript. If nescessary MathScript will be opened first. The text of the function is automatically converted into the MathScript format.

ExportEPS - Exports the text of the selected function as EPS-vector-graphic. This script works only when Math-Script is installed to your system.

Application ARexx commands

The name of the ARexx port of Graph2D is "GRAPH2D.1". If more than one Graph2D's are running, the second, third, ... instance of Graph2D can be referred as "GRAPH2D.2", "GRAPH2D.3" and so on.

Basic commands like "QUIT" exist for every MUI-application and are described in the MUI documentation. In addition Graph2D understands the following very own application commands:

GETFUNCTION PLAIN - returns the text of the function that is actually selected in the function-editor. When there is no selected function you get an empty string "".

GETFUNCTION MS20 - like the option "PLAIN" this one returns the text of the selected function. But this time the text will be converted in a format that is suitable for the program MathScript

1.28 Functions

3.1 Functions

The following chapters are also part of the topic "functions":

3.1.1 Internal operators & functions
3.1.2 Include user functions
3.1.3 EBNF-Syntax

A function in Graph2D has to consist of the following parts:

Name

A function needs a name. The name must not contain leading figures or space and mus not exceed an length of 30 chars. You may use (international) chars, figures (not at first position) and the apostroph "'".

Argument list

The name is followed by an argument list in bracktes "(" ")". The list consists of (many) variable names seperated by colons ",". The variable names must follow the same conventions as the function names. A variable must not appear twice or more in an argument list. The list can also be empty - in this case you do not even have to write the brackets. Name and argument-list together are called "functionheader".

"="

Between functionheader and functiontext you must place an "="...

Function text

The definition of the function-text follows the "usual" rules for the

```
notation of mathematical terms. Graph2D is case-sensitive! The functiontext
may consist of:

  - Operators  see also chapter internal Operators

  - Functions  see also chapter internal Functions

    You also can include other user-defined functions out of the function-
    list into one of your functions.

  - Constantes

    A Constante is e real number in the range \ensuremath{\pm}9.22337177E18

  - Variables

    Of course you can use only those variables in a function you did
    define in the argument-list of the function.


Examples for correct functions:

  "p(x)=x^2+98",
  "Sum(a,b)=a+b",
  "g=9.81",
  "f(x)=sin(x^2)+1/2*x",
  "function23a(iks,yps,zett)=iks^yps*zett+(-1.23+1.0356E-5)"
```

1.29 Internal functions and operators

```
3.1.1 Internal functions and operators


The following functions and operators are by default known to Graph2D:

  sin(x), arcsin(x), sinh(x),
  cos(x), arccos(x), cosh(x),
  tan(x), arctan(x), tanh(x),
  arctanh(x)    - The well-known trigonometrical functions
  abs(x)        - The absolute value of the argument, also known as |x|
  int(x)        - The integer value of the argument
  ln(x)         - natural logarithm (to the base 2)
  sqrt(x)       - The square root x^(1/2)
  sgn(x)        - returns 1, 0, -1 in dependance of x>0, x=0, x<0
  fak(x)        - known as x!. e.g. fak(4)=4*3*2*1=24
                  (only defined for positive integer values)
  if(a, b)      - returns b if a is TRUE (=1) otherwise 0
  binom(n, k)   - binomialcoefficient
  exp(x)        - exponential-function e^x
  e             - 2.71...
  pi            - 3.1415...

  + , -
  * , /
  ^             - I think you know that, don't you? Note: a^b^c = (a^b)^c
```

```
  = , > , <,
  >=, <=, <>   - Compare the two operands
                 e.g. a > b results 1 if a > b otherwise a > b results 0
  AND, OR, NOT - Boolean operators. Only use them for values like
                 1 (TRUE) and 0 (FALSE). Do not expect a bit-wise
                 work of those operators!
```

1.30 Include functions

3.1.2 Include user functions

User defined functions (functions defined by you) can include other user-defined functions into their own definition. If you do that, you should take care of the following things:

Changes to the included function cause no changes to the including function! If you want Graph2D to accept the changes you have to reinterpret the including function.

You absolutely have to avoid (even indirect) circle-definitions! Never do e.g. this: a(x)=c(x); b(x)=a(x); c(x)=b(x)

Examples:

```
  "Sum(a,b)=a+b"
  "f(x)=2^Sum(x,5)+x^2"              correct

  "k=5.6"
  "foobar(x,y)=k*x+y+1"              correct

  "f(x)=g(x)/6"
  "g(x)=2^f(x)"                      error! (circle-definition!)

  "u(x)=u(1/x)+3.14"                 error! (circle-definition!)
```

1.31 EBNF

3.1.3 EBNF-Syntax

Spaces are allowed (except inside of qualifiers/names) in the whole functiontext and may be used to structure the term.

```
  Function   = Qualifier [ "(" [ Qualifier { "," Qualifier } ] ")" ]
               "=" Expression .

  Expression = Prio1 [ ( "=" | ">" | "<" | "<=" | ">=" | "<>" )
               Expression ] .

  Prio1      = [ "+" | "-" ] Prio2 [ ( "+" | "-" | "OR" ) Prio1 ] .
```

```
  Prio2      = Prio3 [ ( "/" | "*" | "AND" ) Prio2 ] .

  Prio3      = [ "NOT" ] Prio4 [ "^" Prio3 ] .

  Prio4      = Number | "(" Expression ")" | Qualifier | Call .

  Call       = Qualifier [ "(" Expression { "," Expression } ")" ] .


  Number     = Figure { Figure } [ "." Figure { Figure } ]
               [ "E" ["-"] Figure { Figure } ] .

  Qualifier  = Letter { Letter | Figure } .

  Figure     = "0" | .. | "9" .

  Letter     = "A" | .. | "Z" | "a" | .. | "z" | "'".
```

1.32 Mui

3.3 MUI

MUI-Copyright:

This application uses

MUI - MagicUserInterface

(c) Copyright 1993/94 by Stefan Stuntz

MUI is a system to generate and maintain graphical user interfaces. With the aid of a preferences program, the user of an application has the ability to customize the outfit according to his personal taste.

MUI is distributed as shareware. To obtain a complete package containing lots of examples and more information about registration please look for a file called "muiXXusr.lha" (XX means the latest version number) on your local bulletin boards or on public domain disks.

If you want to register directly, feel free to send

DM 30.- or US$ 20.-

to

Stefan Stuntz
Eduard-Spranger-Straße 7
80935 München
GERMANY

Comments on MUI concerning Graph2D:

Graph2D needs MUI 2.3 or higher to be installed. You are allowed to use MUI without registering for it - but when you register you can take advantage of some extended functions in the MUI preferences. It is very recommended to read the MUI documentation carefully - especially of the MUI preferences. Despite that i would like to show you in the following list some advantages of MUI-programs that could be useful using Graph2D:

If you want Graph2D to work on an own screen and not on the Workbench then simply configure Graph2D with the MUI preferences to use any screen you like.

Windows of MUI applications are resizeable and completely fontsensitive what means, that they look fine with every font.

MUI applications may be iconified at every time with an extra-gadget in the windows title bar.

MUI applications can optionally completely be handled with the keyboard. Via Tab-cycling and shortcuts every gadget may be (de-)activated without having to use the mouse. The gadget actually receiving keyboard input is always marked with a border or something like this. Windows can be closed normally by pressing ESC.

MUI applications are known to the system as commodities and can so be handled with the commodity-exchange program.

1.33 MathScript

3.4 MathScript

MathScript is a formula-editor by Simon Ihmig. It is under copyright of its Author and it is shareware. MathScript is not part of Graph2D and the Graph2D-package. So if you want to use MathScript you have to look for it yourself (e.g. in the "aminet/misc/math") and pay the registration fee to Simon Ihmig.

Besides many other features MathScripts is able to export mathematical terms as high-quality-EPS-verctor-graphic which for example can be imported by most of the popular word-processors. These results are excellent and much better than the "math. notation" of Graph2D.

To exchange functions between Graph2D and MathScript, Graph2D contains some ARexx-scripts. They were tested for MathScript Version 2.0 and 2.1 and it cannot be guaranteed that they will work with any future Version of MathScript (although they are expected to be).

1.34 SIRDS

3.5 SIRDS

Graph2D uses for the three-dimensional presentation of function a technique known as SIRDS (= Single Image Random Dot Stereogramm). You do not need special glasses or something like this, you may even print out the pictures without loosing the effect.

The only thing you have to do to achieve the 3D-effect ist to look at the pictures in a special way. Instead of looking at the picture itself you have to look behind the picture. Of course you will not see the random dots sharp this way - but this is nescessary for the three-dimensionality.
It may take sometimes until the effect is going to come, and for strange reasons some people never get the "kick". For further instructions how to achieve the effect please refer to the numerous publications concerning SIRDS pictures.

I want to thank Kilian Singer for his small but effective piece of code that makes those pictures possible in Graph2D!

1.35 Index

3.5 Index