Chapter 2
Warm-Up Drills and Tips for Mathematical Software

Tatsuyoshi Hamada

Abstract In Chap. 1, we studied the basic theory of Gröbner bases. Our goal is to use mathematical software to further our research. In this chapter, we will begin with warm-up drills in order to learn the basic ideas necessary for using mathematical software. We will use MathLibre, a mathematical software environment. It is a collection of mathematical software and free documents which form a kind of Live Linux system. The Linux operating system is compatible with UNIX, and many mathematical research systems have been developed on a UNIX system. It is thus important to know the command line interface, Emacs editor, and the fundamental ideas of the UNIX environment. If you are already familiar with this environment, you can skip this chapter; otherwise, please try and enjoy the world of MathLibre.

2.1 Using MathLibre

We now introduce *MathLibre*, a mathematical software execution environment. MathLibre is a kind of *Linux* operating system that boots from a DVD. Linux is a *UNIX*-compatible computer environment used for education and research. MathLibre includes over 100 mathematical software systems that have been developed all over the world. Once MathLibre has been booted, the mathematical software is immediately available for anyone to try.

T. Hamada (✉)
Department of Applied Mathematics, Fukuoka University, Nanakuma, Fukuoka 814-0180, Japan
e-mail: hamada@fukuoka-u.ac.jp

T. Hibi (ed.), *Gröbner Bases: Statistics and Software Systems*,
DOI 10.1007/978-4-431-54574-3_2, © Springer Japan 2013

2.1.1 How to Get MathLibre

In the following, we will assume the computer environment is a PC with a Microsoft Windows operating system. MathLibre [4] is an open-source project that can be downloaded from the Internet.[1] The DVD-R has about four gigabytes of data, and it will take over 30 min to download it. It contains the ISO image file, which, when burned to a DVD, will produce a DVD-bootable version of MathLibre. On your computer, please double-click the icon for the DVD drive; we will find some folders and files in the Explorer window. If there is only one file on the DVD, then it is necessary to reconfigure the DVD burning software and rewrite the ISO image.

2.1.2 How to Boot and Shut Down MathLibre

After successfully making and rebooting the MathLibre DVD, we can find the penguin icon. Press the Enter key and, after a display of the boot-sequence messages, the desktop environment of MathLibre will be displayed. In some cases, it will reboot Windows; if this is the case, the *BIOS* settings need to be reconfigured. When a PC is rebooted, the message "BIOS Setup" is briefly displayed. After pressing the correct function key, usually `<F2>` or `<F8>`, we can find the "Boot" menu in the BIOS configuration. By changing the order of booting, we can boot from a DVD or a USB storage device. If you are not familiar with computers, consult a specialist for help.

If you are using an Apple MacOS X computer with an Intel CPU, you can boot from the DVD by using the "C" key.

2.1.3 Various Mathematical Software Packages

MathLibre includes many mathematical software packages, such as *CoCoA*, *GeoGebra*, *gfan*, *KSEG*, *Macaulay2*, *Maxima*, *Octave*, *Polymake*, *R*, *Risa/Asir*, *Singular*, *surfex*, *Sage*, and others. The applications introduced in this book show only a subset of the mathematical abilities of MathLibre.

Select the Math menu from the start menu at the bottom left-hand side of the screen, as you would do for Windows Start. Alternatively, double click the "Math Software" icon; there is a collection of start-up icons for mathematical software and a "MathLibre Start" button, which leads to an HTML file that contains short introductions and links to the developers of the various software packages.

---

[1] http://www.mathlibre.org/.

2.2 File Manager

In this section, we introduce some of the basic management of files in MathLibre. If you click on the *PCManFM* icon  in the bottom panel (the *lxpanel*), a list of files and folders will be displayed. This is the file manager, *PCManFM*. It is a tool for moving, removing, and duplicating files and folders. It can also be used to start applications (Fig. 2.1).

In the MathLibre Live system, `/home/user` is our *home directory* and `user` is our username. The home directory is a special folder in which we can freely make files and folders. Linux's *directory* and Windows' *folder* have almost the same meaning. In a Linux environment such as MathLibre, the files and folders are in a *tree structure*, such as the one shown in Fig. 2.2. In the figure, the folders are represented by circles. The root of the tree structure is the *root directory*, and it is represented by a slash mark `/`. Unlike Windows, there is no concept of C

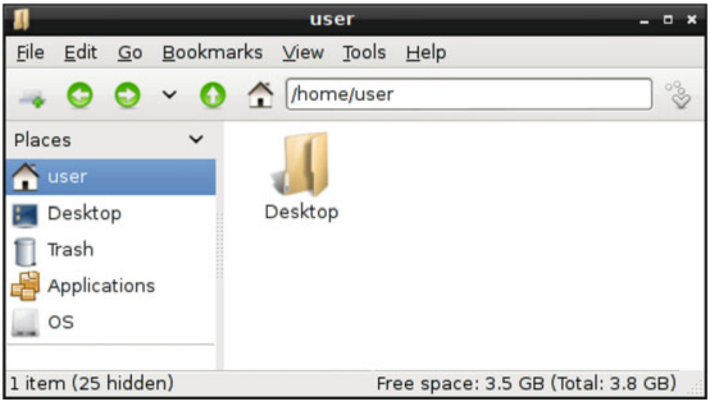


Fig. 2.1 PCManFM

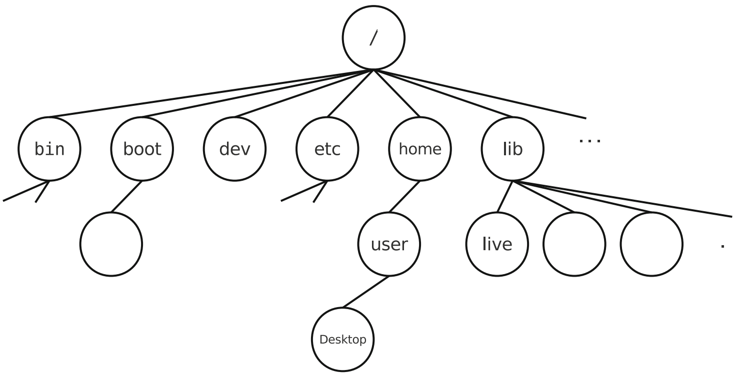


Fig. 2.2 Tree structure of MathLibre

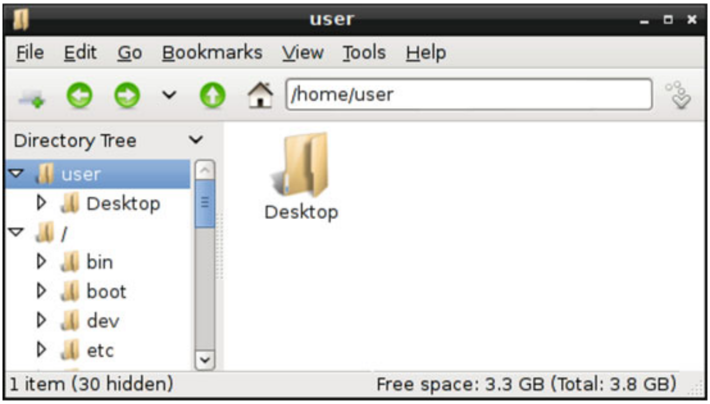


Fig. 2.3 Tree structure of MathLibre with *PCManFM*

or D drives. All files and directories exist under the root directory. There are, however, multiple *subdirectories* branching from the root directory. A slash mark / is used to indicate the path connecting the directory with a subdirectory. This creates the *path of file systems*. According to this rule, the home directory can be seen as a subdirectory `user` of the subdirectory `home` in the root directory `/`. We represent the location of the directory of interest from the root directory. We call this representation the *absolute pathname*. Selecting "View → Side Panel → Directory Tree" of the *PCManFM* menu displays the directory tree structure in the side panel of the *PCManFM* window, as shown in Fig. 2.3.

Usually, the home directory `/home/user` is represented by a tilde ~. When booting from the MathLibre DVD, a home directory is made on the *main memory* of the PC. Please note that all the files in the home directory are removed when the PC is shut down, but you can save your files on a USB flash drive. First, select "Copy" from the menu, then select "View → Side Panel → Place". Next, select the location where you wish to save the file (either the hard disk drive or a USB flash drive), right click, and select "Paste". To move the file, choose "Cut" instead of "Copy". Alternatively, open two windows of *PCManFM*, and then you can drag and drop the target file.

2.2.1 New Folder

To organize work files, you may create a new folder (directory), which is a kind of file. To do so, select "Create New... → Folder:" from the context menu. You will see the message, "Enter a name for the newly created folder:". Type in the desired name and click the OK button. Note that allowable characters for file and folder names include letters, numbers, `.` (period), `-` (hyphenation), and `_` (underline).

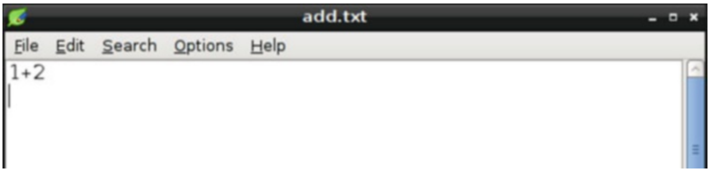


Fig. 2.4 *Leafpad*

2.2.2 *New Text File*

When using mathematical software, it is often convenient to save commands or scripts in a *text file*. A text file is structured as a sequence of lines of electronic text.[2] To create a new text file, in the context menu of *PCManFM*, select "Create New... → Blank File". The message "Enter a name for the newly created file:" will be displayed. Type in the name for the new file and click the OK button; a blank text file with 0 bytes of data will be created.

Double click on the newly created file to launch MathLibre's default *text editor*, *Leafpad*. *Leafpad* is similar to *Notepad* of Windows; it is very simple and can be used easily by anyone. Besides *Leafpad*, MathLibre contains other editors. *Emacs* and *vim* are popular text editors for UNIX users. *Emacs* is not only an editor, but also an environment for developing and computing. A lot of mathematical software uses the *Emacs* interface, and in Sect. 2.6, we will introduce the *Emacs* with MathLibre.

Exercise 2.2.1. Make a text file `add.txt`. As shown in Fig. 2.4, type the characters $1 + 2$ and then click on the "Enter" key. Save this file to the home directory.

2.3 Terminal

Using MathLibre, we can take advantage of mathematical software that has been developed around the world. In order to explore special research software, it is helpful to know how to operate Linux, one of the UNIX-derived systems. It uses a traditional method in which a command is input to the *terminal* by using a keyboard. We will use the *GNOME Terminal* application. To begin, we launch the terminal by single-clicking the third icon.  When we open the terminal, we will see the following:

```
user@debian:~$
```

[2]Text files have an important role in UNIX. If you want to learn more about it, we recommend [2, 3].

We call the phrase "user@debian:~$" the *prompt*. In the prompt, "~" is a special symbol that indicates the home directory /home/user that we created with *PCManFM*. Type a command after the prompt and then press Enter; the command will be executed with the program *shell*. If the command has output, it will be displayed on the terminal.

In this book, we will sometimes omit the prompt and represent it with only $ or #. These prompts represent the general user mode and the system administration mode, respectively. Since the system administration mode is unnecessary for using mathematical software, please work in the general user mode.

2.3.1 Files and Directories

To obtain information about files and directories, use the command ls, which is an abbreviation of the word "list". In the following, we can see the directory Desktop and text file add.txt which were made in an earlier exercise.

```
user@debian:~$ ls
Desktop add.txt
```

On the actual screen, directories are represented in blue and files are shown in white. If we want to know more about a file, for example, its date or size, enter the command with the "long" option: "-l".

```
user@debian:~$ ls -l
drwxr-xr-x 2 user user 4096 2011-01-29 11:14 Desktop
-rw-r--r-- 2 user user    4 2011-01-29 11:14 add.txt
```

To facilitate our work, we will make the working directory in our home directory. The command for making a directory is mkdir, which is an abbreviation of "make directory".

```
user@debian:~$ mkdir tutorial
```

After the command mkdir, type in the new directory name; in this case, it is tutorial. The new name is the *argument* of the command mkdir. It is a distinctive feature of the terminal that there is no message when we have made a new directory. To verify the existence of a directory, use the command ls or *PCManFM*.

```
user@debian:~$ ls
Desktop add.txt tutorial
```

At any time, the directory that we are working with is called the *current working directory*. The current working directory is represented with one period ".". When we launch the terminal, the current working directory is the home directory. The command for changing the current working directory is cd, which is an abbreviation

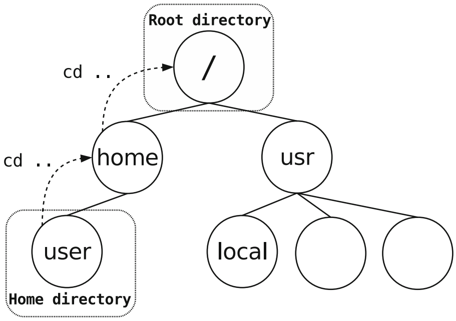


Fig. 2.5 Parent and child directories

of "change directory". After typing in the command, type in the argument (the name of the directory to which you want to move), and then execute the command.

```
user@debian:~$ cd tutorial
user@debian:~/tutorial$
```

It should be noted that the current working directory `~/tutorial` is included in the prompt. `~/tutorial` is a subdirectory of our home directory, and it is also called a *child directory*. If a directory contains a child directory, we call it a *parent directory*, and we represent it with a double period "`..`". Therefore, this command will move from a child directory to its parent directory:

```
user@debian:~/tutorial$ cd ..
user@debian:~$
```

For example, if the starting point, the current working directory, is our home directory `/home/user`, then using the absolute pathname will move us to the directory `/usr/local`:

```
user@debian:~$ cd /usr/local
user@debian:/usr/local$
```

On the other hand, using the symbols for parent directory "`..`" and path `/`, we can represent it like this (Fig. 2.5):

```
user@debian:~$ cd ../../usr/local
user@debian:/usr/local$
```

The location relative to our current working directory is called the `../../usr/local` *relative pathname*. Whether to use the relative pathname or the absolute pathname depends on the situation.

Using the command `cd` with no argument is a way to return quickly to our home directory.

```
user@debian:/usr/local$ cd
user@debian:~$
```

And with "`cd -`", we can quickly go back to the previous directory.

```
user@debian:~/Desktop$ cd
user@debian:~$ cd -
user@debian:~/Desktop$
```

2.3.2 Text Files

There are several commands for displaying text files: `cat`, `more`, `less`, and `lv`. In this section, we introduce `cat` and `less`. If we set the text file name as the argument of `cat`, the contents of the text file are displayed on the terminal. For example, we will display the file which we made in Exercise 2.2.1.

```
user@debian:~$ cat add.txt
1+2
```

There is no problem when displaying a small file like `add.txt`, but if we want to display a large file on the terminal, we will not be able to read it because the contents will be streaming past. It is better to use the command `less` for displaying large files. As an example, here is the file `/etc/passwd`.

```
user@debian:~$ less /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
...
```

The command `less` is useful, and it allows us to choose our position in the contents by using the space key and the cursor. We can search forward with `/` and backward with `?`. To end the display, enter `q`. We can read manuals with the `man` command; this is important when learning Linux commands.

For example, to read the manual description of the command `less`, enter the following:

```
user@debian:~$ man less
```

2.3.3 Input and Output

The command `bc` is a standard calculation tool in Linux. By using `bc` and the text file "add.txt", we can calculate the sum:

```
user@debian:~$ bc < add.txt
3
```

In this situation, the less-than sign < is called a *redirection*. In UNIX systems, a redirection allows us to choose a text file as input. Using the greater-than sign as a redirection > allows us to output the result to a text file.

```
user@debian:~$ bc < add.txt > answer.txt
user@debian:~$ ls
Desktop answer.txt add.txt
user@debian:~$ cat answer.txt
3
```

With the output redirection >, we can use `cat` to make a new file. Note that `^D` means to simultaneously press the D key and the Ctrl key, Ctrl+D. It indicates the end of the input data.

```
user@debian:~$ cat > multi.txt
3*4
^D
user@debian:~$ ls
Desktop answer.txt add.txt multi.txt
user@debian:~$ cat multi.txt
3*4
user@debian:~$ bc < multi.txt
12
```

2.3.4 Character Codes

In the previous subsection, when we input characters to text files, it is recognized as bit data and interpreted using *The American Standard Code for Information Interchange* (*ASCII*), a character encoding system that was originally based on the English alphabet. Using a standard tool of UNIX, `od`, we can examine how the characters are treated in a computer. The command `od` is an abbreviation of "octal dump". Octal is the base-8 number system, which uses the digits 0 to 7; a dump is an exact copy of the data as it is held in the computer. In the following table, we list the binary, octal, decimal, and hexadecimal equivalents for the numbers one to sixteen. The binary number system uses only two symbols, {`0, 1`}. The octal system uses {`0, 1, 2, 3, 4, 5, 6, 7`}, decimal uses {`0, 1, 2, 3, 4, 5, 6, 7, 8, 9`}, and hexadecimal uses {`0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f`}.

Using the command `od` with a qualifier, we can also display the contents of the file in hexadecimal. For example, suppose we wish to build a text file ABC.txt that contains only the three characters "`ABC`" and then observe it with a hexadecimal dump. We can do so, as follows. First, use `cat` to create a text file, using the Enter key and Ctrl+D to finish editing.

Binary	Octal	Decimal	Hexadecimal
0	0	0	0
1	1	1	1
10	2	2	2
11	3	3	3
100	4	4	4
101	5	5	5
110	6	6	6
111	7	7	7
1000	10	8	8
1001	11	9	9
1010	12	10	a
1011	13	11	b
1100	14	12	c
1101	15	13	d
1110	16	14	e
1111	17	15	f
10000	20	16	10
...	...	...	...

```
user@debian:~$ cat > ABC.txt
ABC
^D
```

We can then display the hexadecimal dump by using the command `od`.

```
user@debian:~$ od -Ad -tx1 ABC.txt
000000 41 42 43 0a
000004
```

What is listed here is the information recorded as ASCII code in the storage device. Note that the file name “ABC.txt” is not included in the file itself. The six digits on the left side provide the decimal address of one byte of the stored data. Memory addresses from `000000` up to `000003` are allocated for the data in the file ABC.txt. Since the data is represented in hexadecimal notation, that corresponds to `'A'=41`, `'B'=42`, `'C'=43`. A single two-digit number written in hexadecimal equals 1 byte (= 8 bits). The last two-digit number, `0a`, is the control code for `LF`, which stands for LineFeed. The letters of the English alphabet, numerals, and symbols are each represented by 1 byte in binary. For example, `'A'` is `01000001`; it is represented by eight digits in binary. If we use hexadecimal, `'A'` is `41`. Binary notation is cumbersome for humans to read, so in many cases, data is represented in hexadecimal. One obvious convenience of using hexadecimal is that four digits of binary correspond to a single digit of hexadecimal.

Exercise 2.3.1. Use the command `man ascii` on the terminal to find the ASCII code.

2.4 How to Write Mathematical Documents

When using MathLibre to create a document (such as a research paper) that contains mathematical formulas, we use the *TeX* system. TeX is a typesetting system designed and mostly written by Donald E. Knuth, who is a famous mathematician and computer scientist.

In this book, we will introduce LaTeX, which is widely used in mathematical communities. It was originally written by Leslie Lamport, and the current version is LaTeX 2_{ε}. We can make a PDF file from TeX source code by using the command `pdflatex`. Because it is suitable for the construction of mathematical documents and for structural descriptions, and is not limited to mathematics, it is widely used for writing papers and books. For example, this book was written using LaTeX 2_{ε}.

In order to create a PDF file from LaTeX 2_{ε} source code, we need the following typesetting process:

TeX sourcecode

↓ `pdflatex`

PDF file

In this section, we will use the terminal.

2.4.1 *Writing a TeX Document*

We need a tool with which to create and edit TeX source code. In this case, we will use the *Emacs* text editor, and will launch it from the terminal. If you are not familiar with *Emacs*, you can use another text editor, such as *Leafpad*.

First, we will create a file of TeX source code with the name “sample.tex”. Adding `&` at the end of command allows us to execute *Emacs* as a background job. That is, we can continue to use the terminal while *Emacs* is executing.

```
user@debian:~$ emacs sample.tex &
```

We next write four lines in sample.tex,

```
\documentclass{article}
\begin{document}
Hello LaTeX
\end{document}
```

LaTeX commands always start with backslash \. After editing the file sample.tex, we save the file and exit *Emacs*. For more information on how to use *Emacs*, see Sect. 2.6.

2.4.2 Making a PDF File

A PDF file can be created by typesetting the source file sample.tex with the command `pdflatex`.

```
user@debian:~$ pdflatex sample.tex
This is pdfTeX, Version 3.1415926-2.4-1.40.13
(TeX Live 2012/Debian) restricted \write18 enabled.
...
```

The `pdflatex` command creates three new files: sample.aux, sample.log, and sample.pdf.

To view the PDF file, sample.pdf, we use the command `evince`.

```
user@debian:~$ evince sample.pdf &
```

Only the sentence "Hello LaTeX" will be displayed because it is the only thing between `\begin{document}` and `\end{document}`. We embed the special TEX command in our source code. Note that LATEX is a kind of markup language.

After setting up the printer, we can print our file by using the File menu. Alternatively, you can move the PDF file to the another environment, such as Windows or MacOS.

2.4.3 Brief Introduction to TEX Source Code

The first line of TEX source code, `\documentclass{}`, is the command that reads the settings for the file. For writing a LATEX document, we start with the following command.

Listing 2.1 TEX source code

```
\documentclass[options]{class}
```

In the sample, the document class is *article* and the default paper size is *letterpaper*. We can change the options; the paper can be other sizes, such as *legalpaper* or *a4paper*, and there are other document classes, such as the standard one, *report*, as well as *book*, *letter*, and *slides*. To create a presentation file, use *beamer*, and for a poster session, use *a0poster*. With the appropriate LATEX options, we can create many styles of documents. The lines starting and ending points of the document are `\begin{document}` and `\end{document}`, respectively, and the lines in between are called the document environment.

Listing 2.2 TEX source code

```
\documentclass[options]{class}
\begin{document}
............................
.....creating documents.....
............................
\end{document}
```

This is the basic style of LaTeX 2_ε source code.

With `\begin{}` and `\end{}`, we can create various typesettings.

Listing 2.3 TeX source code

```
\begin{environment name}
...
\end{environment name}
```

There are environment names for many common usages, such as “equation” for mathematical formulas and “itemize” for list structures. For more detail, please refer, for example, to Wikibooks.[3]

2.4.4 *Math Formulas*

LaTeX 2_ε is good for describing mathematical formulas. When we create a mathematical object in TeX source code, we set the beginning and end points in our document. For example, we want to create the following:

Listing 2.4 PDF view

$$x^2 + y^2 + z^2 - 4 = 0 \tag{2.1}$$

We can produce this by using the equation environment, as follows.

Listing 2.5 TeX source code

```
\begin{equation}
  x^2+y^2+z^2-4=0
\end{equation}
```

The equation number is automatically added, unless we add an asterisk *, as shown.

Listing 2.6 TeX source code

```
\begin{equation*}
  x^2+y^2+z^2-4=0
\end{equation*}
```

Or we can abbreviate it:

Listing 2.7 TeX source code

```
\[
  x^2+y^2+z^2-4=0
\]
```

We can use `$$...$$` instead of `\[...\]`, but this is not recommended because it can make it difficult to see the start and end points. Similarly, to write mathematical characters and equations in our documents, we can use `\(` and `\)` or `$...$`, but for the same reason as above, it is better to use `\(` and `\)`.

[3] http://en.wikibooks.org/wiki/LaTeX/.

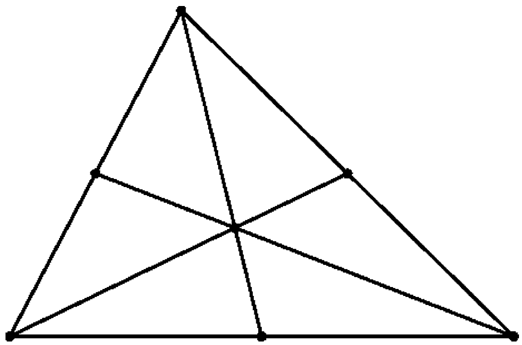


Fig. 2.6 Centroid of a triangle

Listing 2.8 PDF view

Using x_1, x_2, and x_3, we consider the polynomial $x_1^2 + 3x_1x_2 - x_3^2$.

Listing 2.9 TEX source code

```
Using \(x_{1}, x_{2}\), and \(x_{3}\),
we consider the polynomial \(x_{1}^2+3x_{1}x_{2}-x_{3}^2\).
```

Exercise 2.4.1. Edit sample.tex with *Emacs*. Write a formula in the equation environment between `\begin{document}` and `\end{document}`. Next, typeset it with `pdflatex` and then check the PDF file with the `evince` viewer.

2.4.5 graphicx Package

There are some additional packages for LATEX 2_ε, for example, the graphicx package, which lets us embed into our document a graphic file with a PDF, PNG (Portable Network Graphics), or JPEG (Joint Photographic Experts Group) format. As an example, use the following TEX source code Listing 2.10 to embed the graphics file centroid.png.

The command for embedding a graphics file is `\includegraphics[options]{graphics filename}`, and the command `\caption{}` is for naming the graphic file in the document. The *figure* environment is for determining the location of the figure (Fig. 2.6).

Listing 2.10 TEX source code

```
\documentclass{jsarticle}
\usepackage{graphicx}
\begin{document}
......
\begin{figure}[htbp]
\centering
\includegraphics[height=4cm]{centroid.png}
\caption{Centroid of a triangle}
```

```
\end{figure}
......
\end{document}
```

In this section, we presented an introduction to the basic idea of typesetting on the terminal; it is very similar to compiling the source code of a program. In MathLibre, there are many TEX editing environments. We can select from *Kile*,[4] *TeXstudio*,[5] *TeXworks*,[6] and *Texmaker*.[7] There are advantages and disadvantages to each of these environments.

2.5 Various Math Software Systems

There are so many mathematical software systems in MathLibre that it can be confusing. In this section, we deviate slightly from the primary topic of this book because we want you to enjoy mathematical software. We therefore introduce dynamic geometry software, which allows us to create and manipulate geometric constructions with a simulated compass and ruler. It is very popular for educational use.

When I first encountered this, I misunderstood and thought that it was a tool for only elementary geometry. However, as I used the software to create geometrical objects, I began to see interesting applications for the function of drawing trajectories and the construction of recursive methods. I believe that this software has potential for helping us visualize various mathematical ideas. In MathLibre, there is a lot of dynamic geometry software because I like it. One of them includes the automatic proof assistant system that uses the method of Wu and Gröbner bases. This is not covered in this book, but if you are interested in this topic, you can refer to [1].

2.5.1 KSEG

One of the basic dynamic geometry software systems is *KSEG*.[8] This open-source software was written in the C++ programming language by Ilya Baran. Using *KSEG*, we can deform, rotate, and move geometrical objects while maintaining their properties. We can measure the distance between two points and the angles of a triangle, and then perform calculations with them. With a function for creating the locus of restricted objects, we can draw various geometric curves. We can make Koch and dragon curves by using a recursive method.

---

[4] http://kile.sourceforge.net/.

[5] http://texstudio.sourceforge.net.

[6] http://www.tug.org/texworks/.

[7] http://www.xmlmath.net/texmaker/.

[8] http://www.mit.edu/~ibaran/kseg.html.

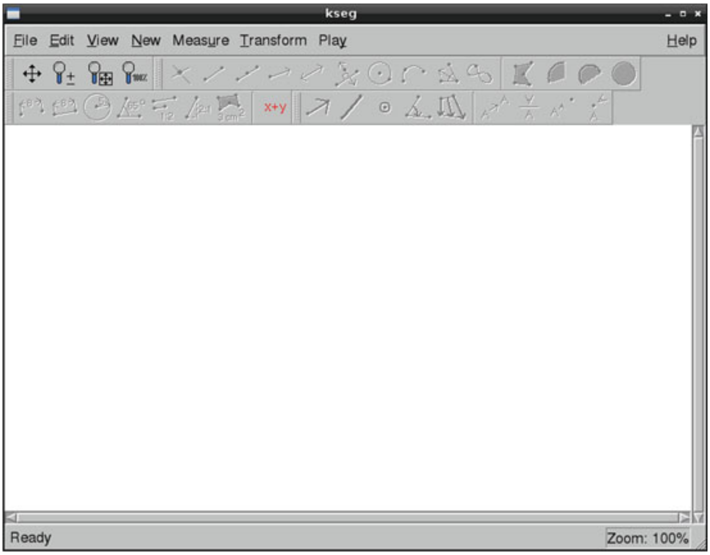


Fig. 2.7 *KSEG* window

Fig. 2.8 *KSEG* menu

2.5.1.1 How to Start KSEG

From the start menu, click on the Math software submenu and select KSeg . Alternatively, to launch from the terminal, enter `kseg`. The following *KSEG* window will be displayed.

There are menus at the top of the window, and button icons right below them. The pictures on the icon buttons indicate their functions (Figs. 2.7 and 2.8).

KSEG has only four main types of function.

1. Draw a point by right clicking.
2. Select points, lines, and circles by left clicking. (They can be selected by using the shift key and rectangle selection.)
3. Create a geometrical object by using a menu or button.
4. Delete geometrical objects using the Ctrl+Del keys.

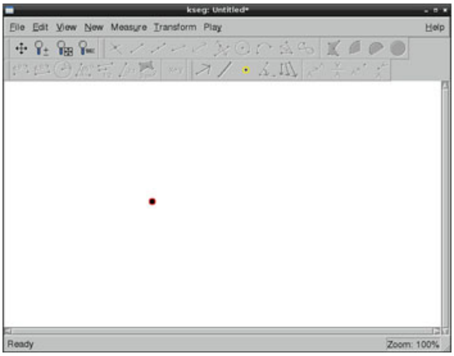


Fig. 2.9 Right clicking the appropriate position

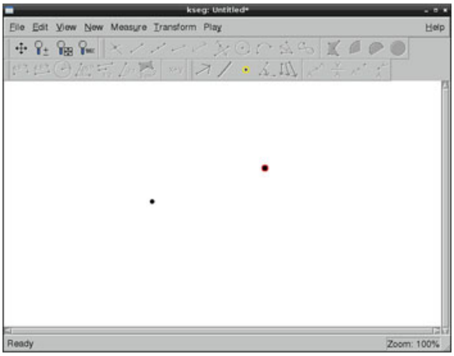


Fig. 2.10 Right clicking the other place

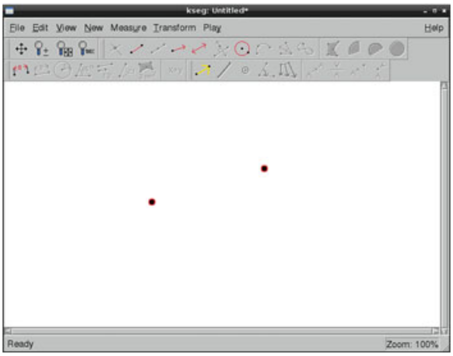


Fig. 2.11 Selecting two points with Shift key

For example, we need two points in order to draw a segment. To create them, right click on two places on the *KSEG* screen. After selecting these two points, create a segment using the menu "New → Segment" or the button of "Segment", Figs. 2.9–2.12.

In a similar way, we can create a "Line" or a "Half line". If we select two points and click the "Circle" button, we create a circle centered at the first point and going through the second point, Figs. 2.13–2.16.

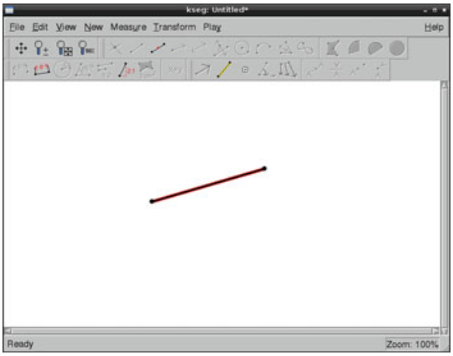


Fig. 2.12 Selecting "New → Segment"

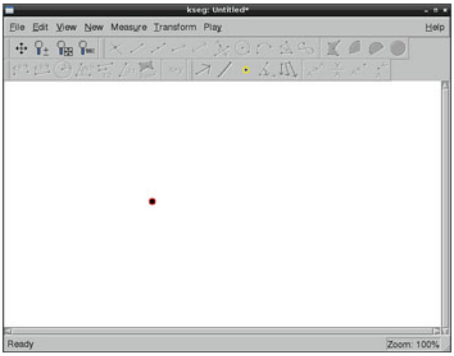


Fig. 2.13 Right clicking the appropriate position

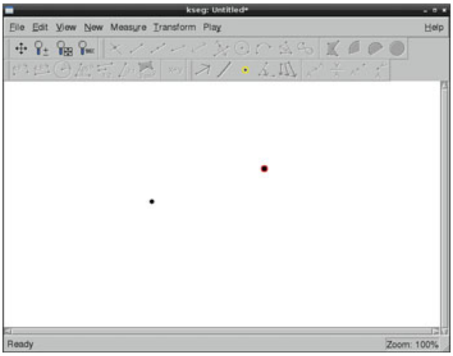


Fig. 2.14 Right clicking the other place

2.5.1.2 Creating a Triangle

Creating three points and using rectangle selection is a convenient way to create a triangle, Figs. 2.17–2.20.

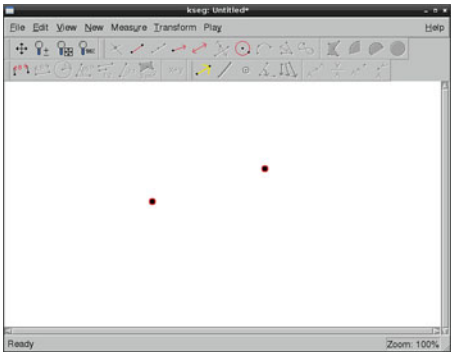


Fig. 2.15 Selecting two points with the Shift key

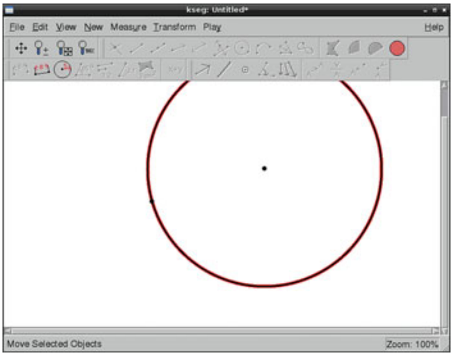


Fig. 2.16 Selecting "New → Circle"

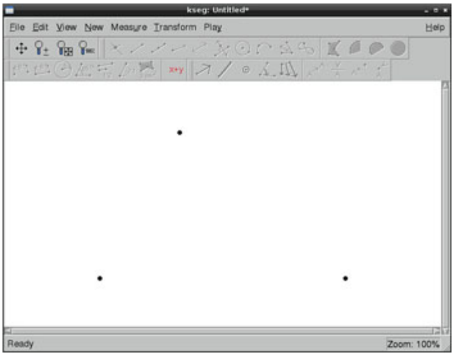


Fig. 2.17 Drawing three points for creating a triangle

2.5.1.3 Centroid of a Triangle

We can find the locations of the various centers of a triangle with *KSEG*. In this subsection, we will explain how to draw the centroid. After we draw the centroid, we can drag the points of the triangle to see its dynamic deformation, Figs. 2.21–2.36. Determining the location of the circumcenter, the orthocenter, and the centers of the incircle and excircle of a triangle are left as exercises for the reader.

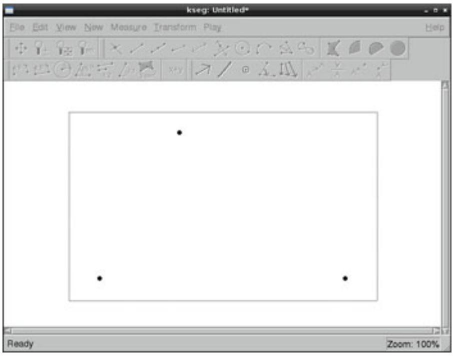


Fig. 2.18 Dragging the mouse around the three points

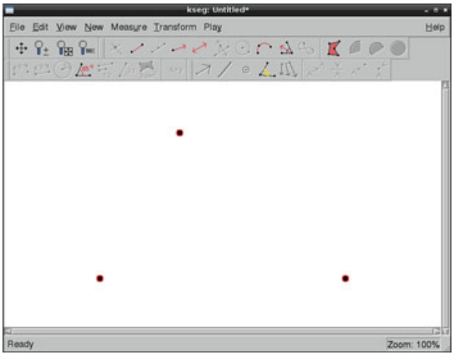


Fig. 2.19 Three points are selected

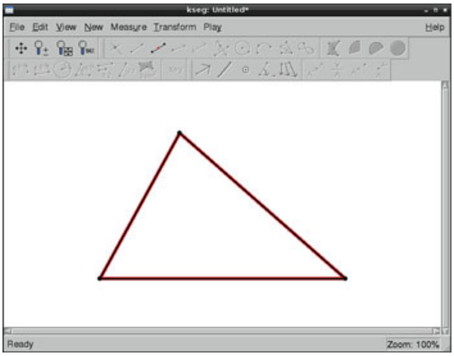


Fig. 2.20 Selecting "New → Segment"

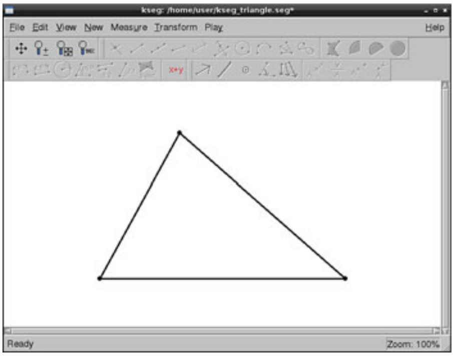


Fig. 2.21 Drawing a triangle

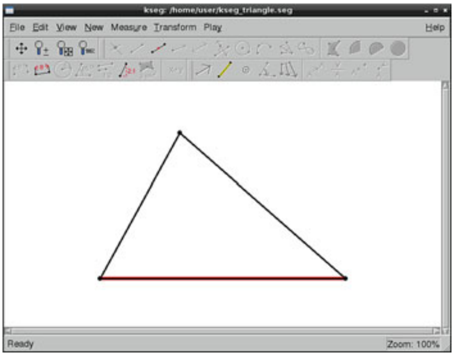


Fig. 2.22 Selecting an edge of the triangle

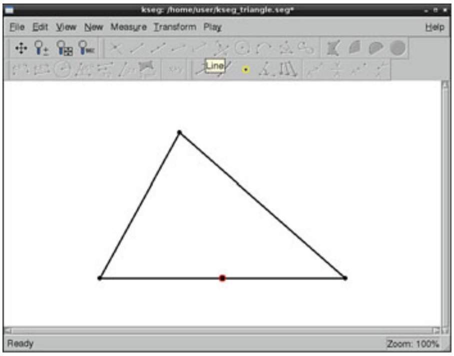


Fig. 2.23 "New → Midpoint"

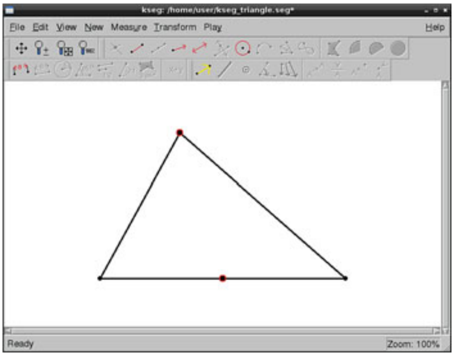


Fig. 2.24 Selecting a vertex and the midpoint with the Shift key

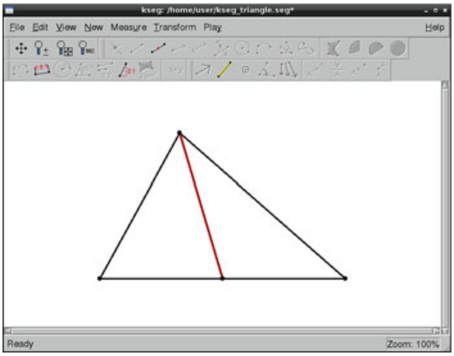


Fig. 2.25 Selecting "New → Segment"

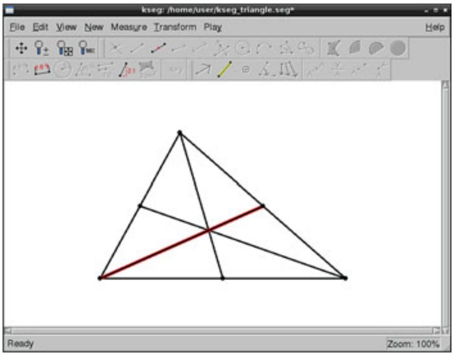


Fig. 2.26 After the same step for the other two points

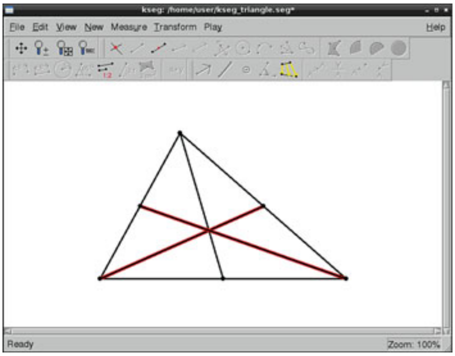


Fig. 2.27 Selecting two medians in the triangle

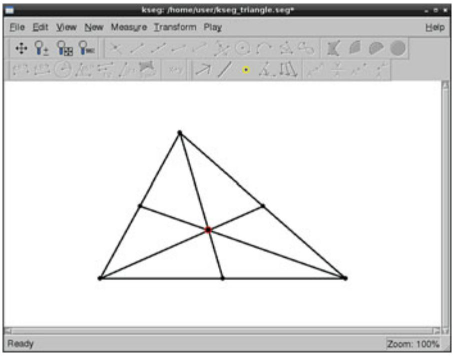


Fig. 2.28 Selecting "New → Crosspoint"

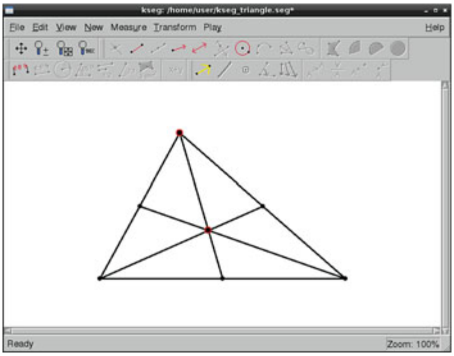


Fig. 2.29 Selecting a vertex and centroid

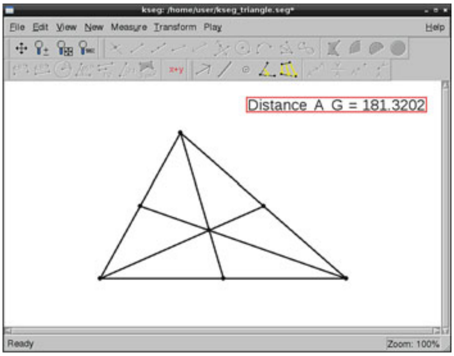


Fig. 2.30 "Measure → Distance"

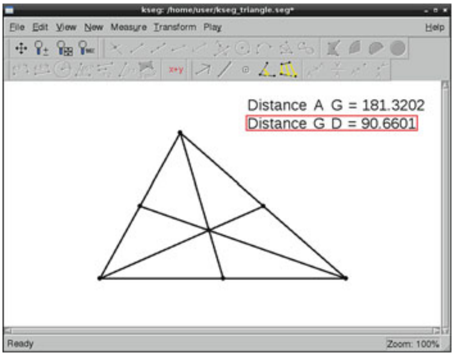


Fig. 2.31 Measuring the centroid and the midpoint

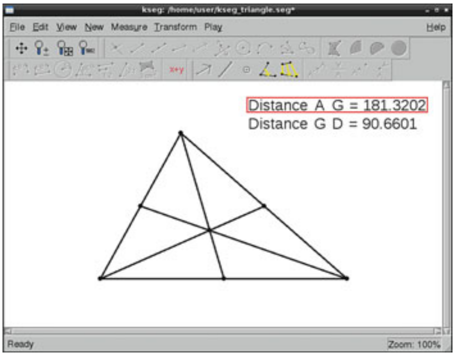


Fig. 2.32 Selecting the longer one

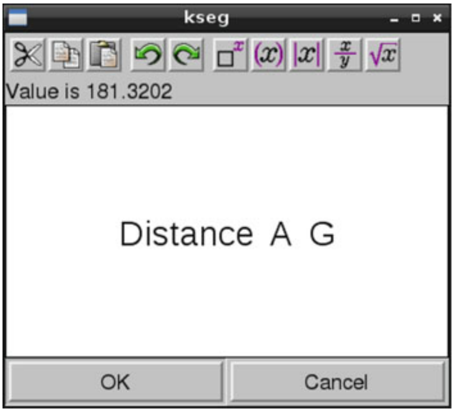


Fig. 2.33 "Measure → Calculate"

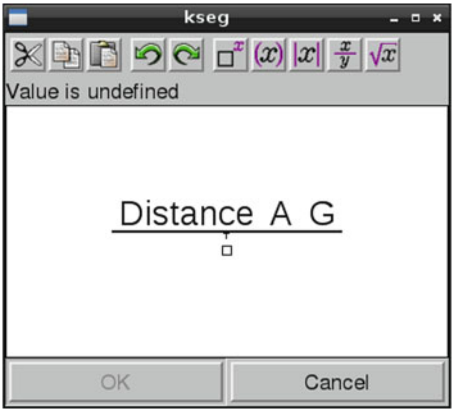


Fig. 2.34 Moving the cursor to the end of line, pushing $\frac{x}{y}$

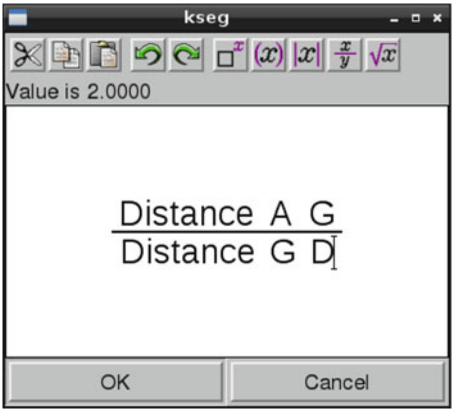


Fig. 2.35 Selecting the shorter one

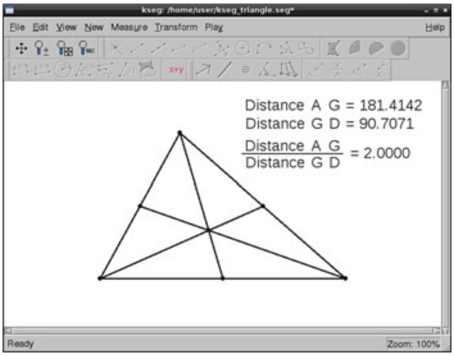


Fig. 2.36 Clicking "OK"

The centroid is exactly two-thirds of the way along each median. We can check this property with the function "Measure".

For more information, see the *KSEG* help document. There are samples on the *KSEG* web page and on the MathLibre DVD. In the *KSEG* help file, go to "File → Copy as Construction" to find out how to construct geometrical objects.

Exercise 2.5.1. Use *KSEG* to draw the centers of a triangle.

Exercise 2.5.2. Consider several ways to use *KSEG* to draw conic sections.

Exercise 2.5.3. Use *KSEG* to draw Koch curves.

KSEG supports various formats for exporting graphics. From the menu, choose "File → Export to Image"; you can then choose one of the following formats: BMP, JPEG, PBM, PGM, PNG, PPM, XBM, and XPM. PDFLaTeX supports PNG and JPEG formats.

2.5.2 GeoGebra

KSEG is excellent for drawing geometrical objects, but it does not support drawing the graphs of functions. We therefore introduce *GeoGebra*, dynamic mathematical software that can draw graphs. *GeoGebra* is open source and was first developed by Markus Hohenwarter when he was a graduate student at the University of Salzburg. He is now a professor at the University of Linz. *GeoGebra* was developed by an international team, and it has became popular all over the world. Much mathematical software is written in the C or C++ language, but *GeoGebra* was developed in Java. In general, software written in Java has the advantage of being easy to move to a variety of environments, such as Windows, Mac, and UNIX. *GeoGebra* is mathematical education software, and so this flexibility is important.[9] In this section, we will discuss the basic operations and investigate the trajectories of the vertex of a parabola.

[9] A disadvantage is that software developed in Java may be slower than that developed in C or C++.

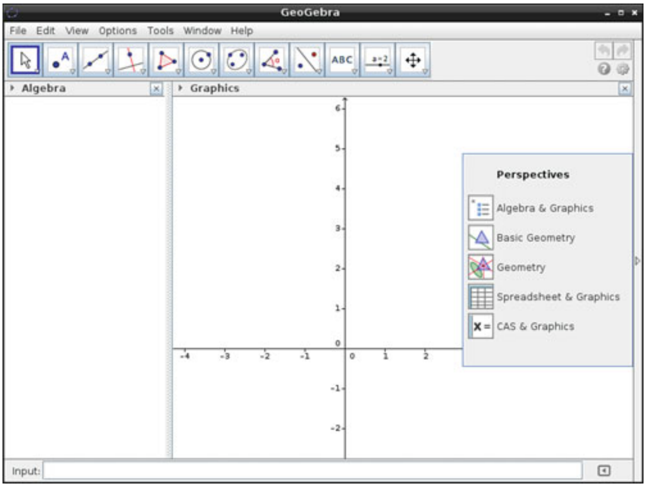


Fig. 2.37 *GeoGebra* window

2.5.2.1 GeoGebra Basics

To launch *GeoGebra*, select the icon from the start menu. To execute it from the terminal, enter the command `geogebra`. With *KSEG*, we first create the points and the select the geometrical object to be created from a menu. With *GeoGebra*; however, we first select the geometrical object and then change to drawing mode. For example, if we want to draw a point in the *GeoGebra* window, we first click the icon for Point . We then create a new point by clicking the appropriate position on the window. To create a circle, first chose the Circle icon . There are four methods for drawing a circle: "Circle with Center through Point" (the default), "Circle with Center and Radius", "Compass", and "Circle through Three Points". Select one of these methods by clicking the small triangle in the right bottom of the icon. There are some differences between the interfaces of *KSEG* and *GeoGebra*, but both of them are good software systems for dynamic geometry (Fig. 2.37).

We can enter a command to the Input Bar, draw the graph of a function, draw the tangent line, or calculate the integral.

First, though, we will draw some points, lines, triangles, and circles. Click the "Move" icon, and then you can use the cursor to freely move the geometrical objects. When we move a circle, the changes in the equation of the circle are displayed on the left side of the window Fig. 2.38.

2.5.2.2 Graph of a Function

We can draw a graph by typing the function into the Input bar of *GeoGebra*. Type in `y=x^2` and press the Enter key to draw the parabola $y = x^2$.

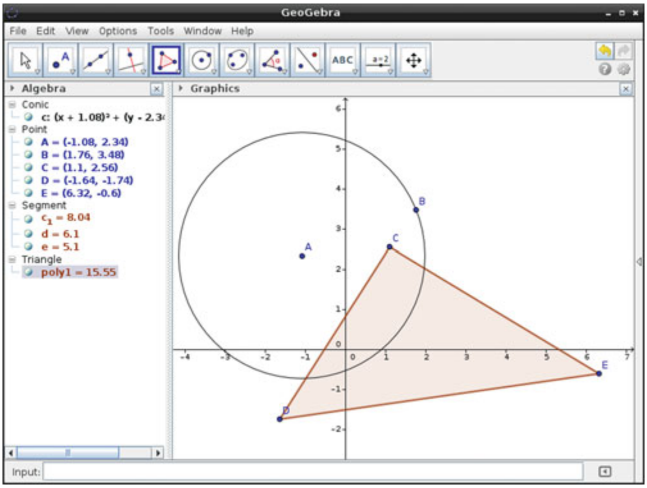


Fig. 2.38 Geometrical object with *GeoGebra*

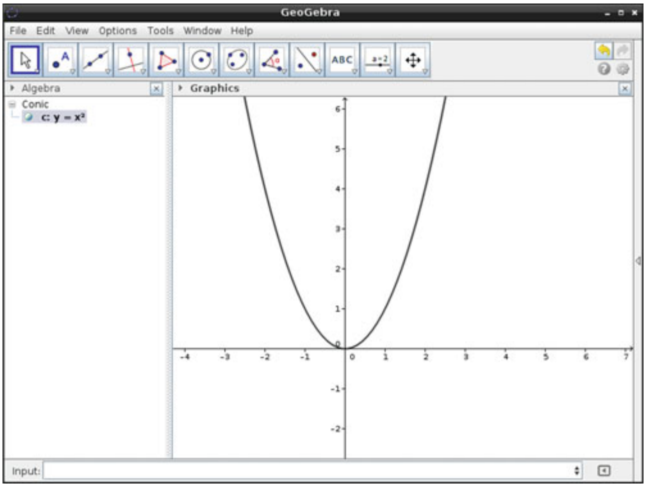


Fig. 2.39 Parabola with *GeoGebra*

The parabola is automatically named c by *GeoGebra*. The function `Vertex[]` will show the vertex of the conic. The argument of the `Vertex[]` function is the name of the conic. In this case, we input `Vertex[c]` and press Enter. The vertex named A is on the origin $(0, 0)$, and by clicking the cursor icon, we can move the parabola. As we move it, we can observe the changes in the vertex A and the equation c of the parabola (Fig. 2.39).

2.5.2.3 Slider of GeoGebra

Suppose we have the following problem: "What is the trajectory of the vertex A of the parabola $y = x^2 - 2ax + 1$ when we change the value of the constant a?"

Slider is a convenient tool for solving this problem.

1. Click the icon of "Slider".
2. Click the appropriate point of the graphics area.
3. The small new window shows the default data of the Slider function; click the Apply button.
4. Type `y=x^2-2*a*x+1` in the Input Bar and press Enter.
5. Type `Vertex[c]` in the Input Bar and again press Enter.
6. Click the Move icon and change the value of the slider.
7. Observe the vertex.

2.5.2.4 Trace On

To follow the changes in the vertex A, we can use a feature called "Trace On".

1. Right click the vertex A.
2. Select the check box of "Trace On".
3. Move the slider and observe the state of the vertex A.

Note that the trace is also a parabola.

2.5.2.5 Creating a Graphics File

GeoGebra also supports exporting to various file formats. As an example, this is how to export to a PNG file (Fig. 2.40):

1. Create a figure with *GeoGebra*.
2. Select from the menu File → Export → Graphic View as Picture (png, eps)...
3. Select Portable Network Graphics (png).
4. Press the Save button.
5. Save the file with a suitable name to an appropriate directory.

GeoGebra supports exporting as a dynamic web page in Java Applet and Javascript. There is a community site for collecting educational materials for *GeoGebra*: GeoGebraTube.[10] It is easy to upload files from the application to this archive. *GeoGebra* also interfaces with the TEX system. It is able to export PGF/Tikz source code; this is a TEX macro package for embedding graphics into TEX source code.

[10] http://www.geogebratube.org/.

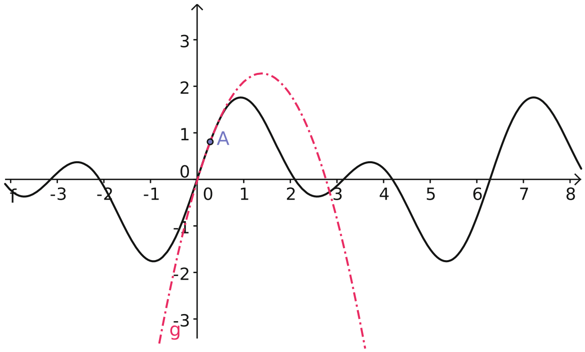


Fig. 2.40 Taylor Polynomial with *GeoGebra*

By typesetting with the command `pdflatex`, we can produce a document with embedded graphics.

Exercise 2.5.4. Using *GeoGebra*, draw a graph of the function $f(x) = \sin(x) + \sin(2x)$.

Exercise 2.5.5. Select a point A on the graph of the function $f(x) = \sin(x) + \sin(2x)$, and draw a graph of the second-degree Taylor polynomial on A.

Exercise 2.5.6. Make a slider of $1 \leq a \leq 5$ with the increment equal to 1. Draw a graph of the Taylor polynomial on A with degree a.

2.5.2.6 GeoGebra with Risa/Asir

By combining *GeoGebra* with other software, we can display the result of a Gröbner basis. Consider the polynomials $x^2 + y^2 - a$ and $xy - b$ of the two variables x and y. In this case, $a, b \in R$ are constants. We can draw the graph of the implicit function. We can create a and b with Slider, and we can choose the range of default values of a and b. After making the two sliders, input the equation `x^2+y^2-a=0`. It is a circle, and we can change its radius. We can input `x*y-b=0`, a hyperbola. Click on the icon for Intersect Two Objects, and select the circle and the hyperbola. When the default value of the sliders are $a = 1, b = 1$, the circle and the hyperbola do not intersect. The result of this operation is “A undefined” because the solution of this system of equations is a complex number. If we change the value so that $a = 2$, there are two intersects, and when a is greater than 2, then are four cross points, which are displayed as real solutions.

Set $a = 4$. *GeoGebra* supports rounding, so select “Option → Rounding → 4 Decimal Places”. The coordinates of the four intersection points are $(0.5176, 1.9319), (-0.5176, -1.9319), (-1.9319, -0.5176)$, and $(1.9319, 0.5176)$ (Fig. 2.41).

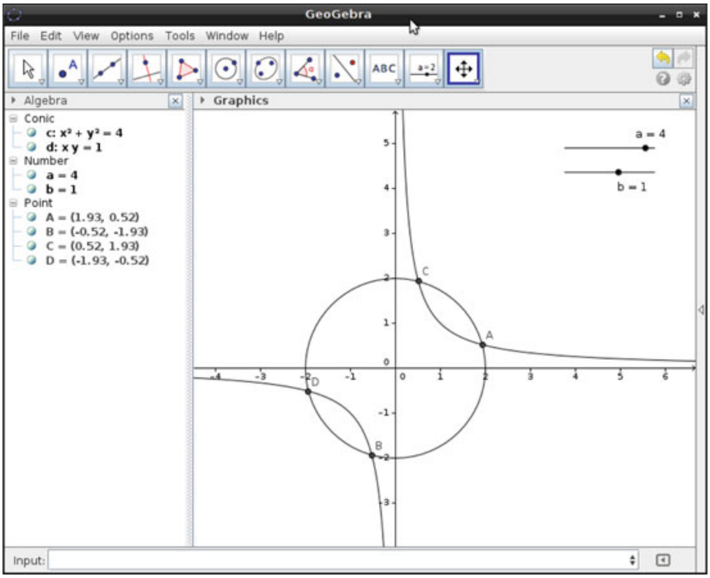


Fig. 2.41 Four intersection points with *GeoGebra*

We will calculate the Gröbner basis by using a computer algebra system, *Risa/Asir*.[11] It is launched in a similar way to the previous programs. To execute it from a terminal, enter `openxm fep asir`. To learn more about the uses of *Risa/Asir*, please see the "Risa/Asir Drill Book"[6].

As an example, here are the commands `nd_gr` and `pari`, in *Risa/Asir*.

```
[1371] G=nd_gr([x^2+y^2-a,x*y-b],[x,y],0,2);
[y^4-a*y^2+b^2,-b*x-y^3+a*y]
[1372] pari(roots,base_replace(G[0],[[a,4],[b,1]]));
[ -1.931851652578136573 -0.5176380902050415246
0.5176380902050415246 1.931851652578136573 ]
```

Using the command

`nd_gr(Polynomial List, Variable List, P, Order),`

we can compute the Gröbner basis of `G[0]=y^4-a*y^2+b^2`, `G[1]=-b*x-y^3+a*y`. In this example, the argument for Order is `2`, which is the lexicographical order (refer Corollary 1.4.2). With the command `pari(Roots, polynomial)`, we can calculate the roots of the polynomial. In this case, we solved the polynomial `G[0]` with $a = 4, b = 1$. This result gives the y coordinates of the four intersects. We see it nearly coincides with the results of *GeoGebra*.

[11] http://www.math.kobe-u.ac.jp/Asir/asir.html.

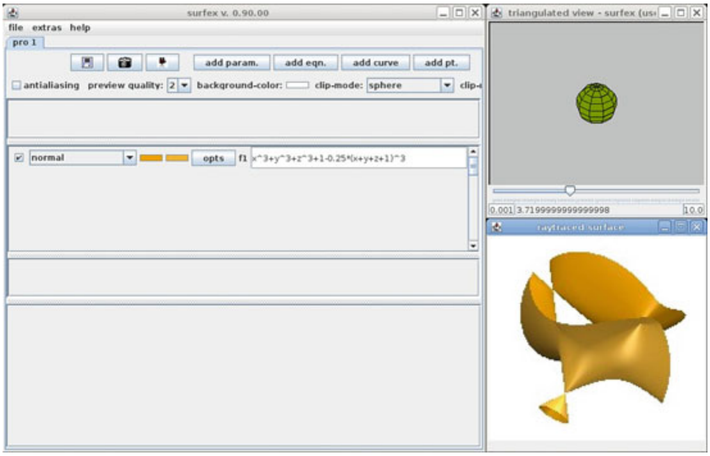


Fig. 2.42 *surfex*

To see this Gröbner basis, we only need to input the result, `y^4-a*y^2+b^2=0` and `-b*x-y^3+a*y=0`, to the Input Bar of *GeoGebra*.

We can draw it with the function `ifplot` of *Risa/Asir*. The commands *surf* and *Sage* are able to draw implicit functions.

Exercise 2.5.7. Let $f(x, y)$ be a polynomial of degree n, and let $g(x, y)$ be a polynomial of degree m. Assume that f and g are relatively prime. At most, how many intersections of $f = g = 0$ are there? Try to determine the number with the help of Geogebra.[12]

2.5.3 *Surf Family*

surf was written by Stephan Endrass. It is a tool for drawing real algebraic geometry. With it, we can create beautiful graphics of plane algebraic curves, algebraic surfaces, and hyperplane sections of surfaces. *surf* supports a macro language that is very similar to the C language, but *surfex* and *surfer* have been released, and they have interactive interfaces. Using *surfex* and *surfer*, we can observe the graphics of surfaces from a dynamically changing viewpoint. All of these are included in MathLibre, but we will introduce *surfex*. When *surfex* is launched, these windows are displayed: The main window has the following four buttons (Fig. 2.42):

[12] Answer: mn (Bezout theorem).

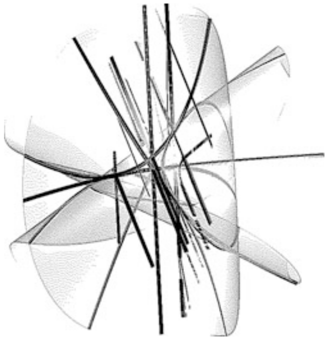


Fig. 2.43
Figure 2.45 + Fig. 2.46

add param. creating a parameter slider,
add eqn. creating a surface that represents an implicit function,
add curve creating a curve as an intersection of surfaces,
add pt. creating a point.

There is already an example polynomial $x^3 + y^3 + z^3 + 1 - 0.25(x + y + z + 1)^3$, which is called the Cayley cubic, and its surface is displayed in the small window. There are two small windows, one is the “triangulated view” and the other is the “raytraced surface”. To change the viewpoint of the display, drag on image in the “triangulated view” window. We can change the colors of surfaces by using the “opts” button. There is also a transparency mode, which is very helpful for observing the intersection of surfaces. The resolution of the graphic can be changed by configuring the parameter of “preview quality” in the main window; the best graphics are when the value is set to 1. There are some buttons in the main window, one of which is the camera icon, which captures the graphics in JPEG format. Using *surfex* with Example 1.4.11 from the first section, we create the following pictures. Because of the difference in the dimensions, it is little bit difficult to see with printed pictures, but we note Fig. 2.43 is similar to Fig. 2.44. All the surfaces in the figure are created by the original polynomials; to reduce complexity, the surfaces from the Gröbner basis have been set to transparent (Figs. 2.45 and 2.46).

2.5.4 Maxima

In this section, we introduce a general purpose system for computer algebra, *Maxima*. *Maxima* is the descendant of *MACSYMA*, which has a long history and is written in the Lisp language. MACSYMA was developed at MIT for a research project on artificial intelligence. In this book, we also introduce other research systems: *CoCoA*, *Macaulay2*, *Risa/Asir*, and *Singular*. These recent systems have been developed mainly for mathematics research. In the 1970s and 1980s, *MACSYMA* supported the Risch algorithm for indefinite integrals; it was commercialized by the company Symbolics [5]. *Maxima* was developed by William Schelter and

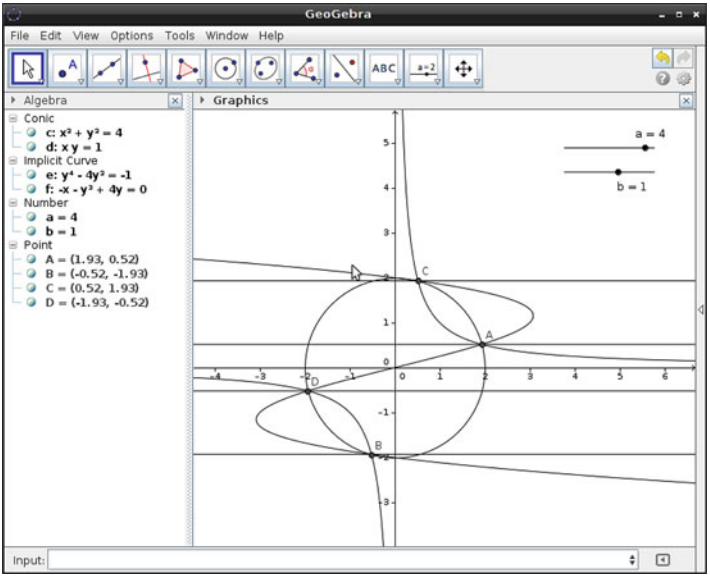


Fig. 2.44 Implicit functions with *GeoGebra*

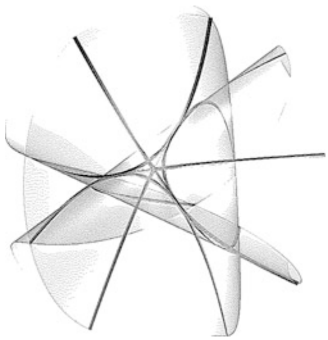


Fig. 2.45 Intersection curves with the original polynomials

is based on a 1982 version of *MACSYMA*; it later became open source. In 2001, Schelter passed away while traveling in Russia, but *Maxima* is now continuously maintained by a team of developers.[13] *Maxima* supports many operations, including factoring; solving algebraic, differential, and integral equations; manipulating limits, series, matrices; and drawing graphs.

We can launch *Maxima* either from the menu or by entering `maxima`, `xmaxima`, or `wxmaxima` to the terminal. As shown here, we launch *Maxima* from the terminal.

[13] http://maxima.sourceforge.net/.

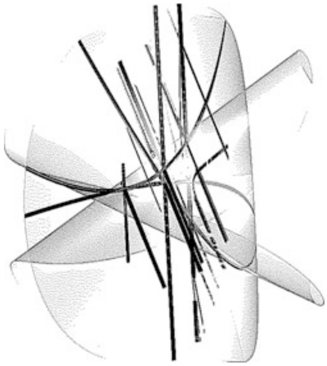


Fig. 2.46 Intersection curves with Gröbner basis

```
user@debian:~$ maxima

Maxima 5.27.0 \url{http://maxima.sourceforge.net}
using Lisp GNU Common Lisp (GCL) GCL 2.6.7 (a.k.a. GCL)
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
(%i1)
```

`(%i1)` is the interactive prompt for *Maxima*, after which a command or function can be entered, followed by a semicolon. To quit the program, enter `quit();`

When learning *Maxima*, `describe()` is an important function. For example, entering the command `describe(factor);` will display the help documents for the function `factor()`. The command `describe(string)` is equivalent to `describe(string, exact)`. If such an item exists, it will find one with the exact same title (case-insensitive) as `string`. The command `describe(string, inexact)` finds all help documents for items that contain `string` in their titles. Note that following the interactive prompt with `? foo` (with a space between `?` and `foo`) is equivalent to `describe(foo, exact)`, and `?? foo` is equivalent to `describe(foo, inexact)`. There is a lot of documentation for *Maxima* in MathLibre and on the Internet.

2.5.4.1 Output of TEX Source Code

TEX is an excellent typesetting system, but it can be difficult for complicated mathematical formulas. Thus, it would be helpful to have mathematical software produce output that is formatted as TEX source code. In fact, there are several mathematical software systems that can do this; in this section, we will show how to do it using the function `tex()` in *Maxima* (Fig. 2.47).

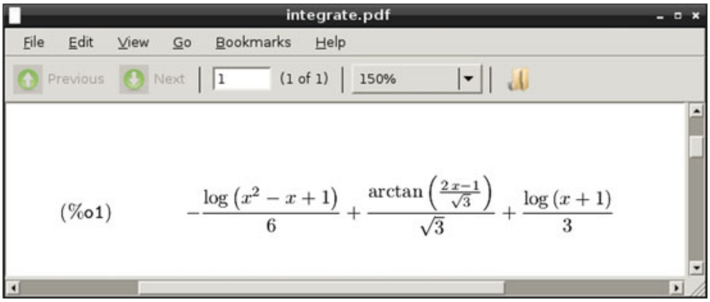


Fig. 2.47 Displayed with evince

Listing 2.11 `tex()` of *Maxima*

```
(%i1) integrate(1/(x^3+1),x);
                                                  2 x - 1
                        2              atan(-------)
                   log(x   - x + 1)        sqrt(3)     log(x + 1)
(%o1)            - --------------- + ------------- + ----------
                          6              sqrt(3)          3
(%i2) tex(%o1);
$$-{{\log \left(x^2-x+1\right)}\over{6}}+{{\arctan \left({{2\,x-1
 }\over{\sqrt{3}}}\right)}\over{\sqrt{3}}}+{{\log \left(x+1\right)
 }\over{3}}\leqno{\tt (\%o1)}$$
(%o2)                                (\%o1)
(%i3)
```

Inputting the command after *Maxima*'s interactive prompt `(%i1)`, allows us to calculate the indefinite integral of the function $f(x) = 1/(x^3 + 1)$. The result of the computation can be referenced to as `(%o number)`.

2.5.4.2 Working Record of a Calculation

We can save the result of a computation by using redirection, but we can also save the entire work record to a text file by using the UNIX command `script` on the terminal. In the following example, we use *Maxima*, but the method is the same for any other mathematical software which is executed from the terminal.

```
user@debian:~$ script
Script started, file is typescript
user@debian:~$ maxima

Maxima 5.27.0 \url{http://maxima.sourceforge.net}
using Lisp GNU Common Lisp (GCL) GCL 2.6.7 (a.k.a. GCL)
Distributed under the GNU Public License. See the file COPYING.
Dedicated to the memory of William Schelter.
The function bug_report() provides bug reporting information.
(%i1) integrate(1/(x^3+1),x);
                                                  2 x - 1
                        2              atan(-------)
```

```
                     log(x  - x + 1)          sqrt(3)     log(x + 1)
(%o1)              - ---------------- + ------------- + ----------
                            6              sqrt(3)          3
(%i2) quit();
user@debian:~$ exit
Script done, file is typescript
user@debian:~$ ls
Desktop typescript
user@debian:~$
```

When we execute the command `script` with no arguments, the working record is saved to the file `typescript`. If we specify the file name in the argument as follows, it will be saved in the specified file.

```
user@debian:~$ script logfile.txt
Script started, file is logfile.txt
```

The text editor *Emacs* can be used to view the working record and to alter it to try different approaches (ref. Sect. 2.6).

2.5.5 R

R is a programming language and environment for statistics and graphics.[14] It is very similar to the *S* language and statistical calculation environment, although *R* and *S* were developed independently. *R* is open source and has grammar similar to the *S* language. To execute *R*, enter `R` to the terminal.

```
user@debian:~$ R

R version 2.15.1 (2012-06-22) -- "Roasted Marshmallows"
Copyright (C) 2012 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
We are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

[14] http://www.r-project.org/.

The interactive prompt for *R* is `>`. There are various ways to execute *R*, for example *Rcommander*, *Rkward*, and *RStudio*. They can be found on the start menu or by entering the command `Rcmdr`, `rkward`, or `rstudio`. *R* can also be executed in the text editor *Emacs*, as with other mathematical software systems.

2.5.6 Sage

Sage is a free open-source mathematics software system. It combines the power of many existing open-source packages, such as *Maxima*, *PARI*, *R*, *Singular*, and *surf*, into a common Python-based interface. The lead developer of *Sage* is William Stein, a professor at the University of Washington. There are large communities of *Sage* developers and users all over the world. This software supports a huge range of mathematics, including basic algebra, calculus, from elementary to very advanced number theory, cryptography, numerical computation, commutative algebra, group theory, combinatorics, graph theory, exact linear algebra, and much more. To run *Sage* on MathLibre, select "Math → SAGE" from the start menu. The message shown below will appear in the new terminal window. The *notebook* is a browser-based interface for *Sage*, and in this system, it will start automatically.

```
----------------------------------------------------------------------
| Sage Version 5.7, Release Date: 2013-02-19                         |
| Type "notebook()" for the browser-based notebook interface.        |
| Type "help()" for help.                                            |
----------------------------------------------------------------------

Please wait while the {\it Sage Notebook\/} server starts...
Setting permissions of DOT_SAGE directory so only you can read and write it.
The {\it notebook\/} files are stored in: sage_notebook.sagenb
```

Before proceeding further, it is necessary to enter and confirm a password.

```
Please choose a new password for the {\it Sage}\/ Notebook 'admin' user.
Do _not_ choose a stupid password, since anybody who could guess our password
and connect to our machine could access or delete our files.
NOTE: Only the hash of the password you type is stored by {\it Sage}.
You can change our password by typing notebook(reset=True).

Enter new password:
Retype new password:
```

A new window of the *Iceweasel*[15] web browser will open with *Sage notebook* (Fig. 2.48). We can create a new worksheet by clicking on "New Worksheet" (Fig. 2.49).

For example, we can change the name of a worksheet; here, it is "ex1". We can then integrate a function. When we click the check box of "Typeset", the typeset formula in Fig. 2.50 will be displayed.

To see how to draw the curve of an implicit function, we can find an example in the Help file. We can use *Singular* and *surf* for to plot a curve using *Sage* (Fig. 2.51).

[15]Iceweasel is *Firefox*, rebranded.

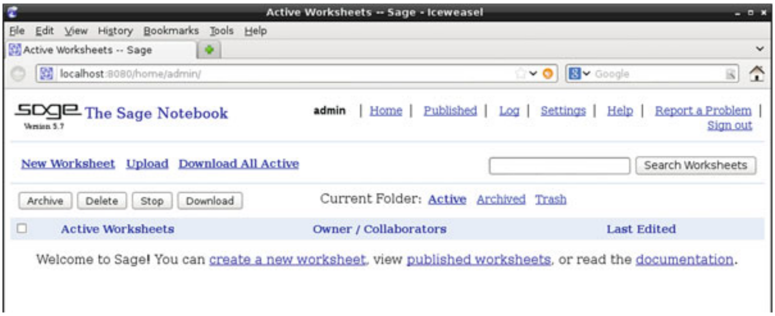


Fig. 2.48 *Sage* notebook

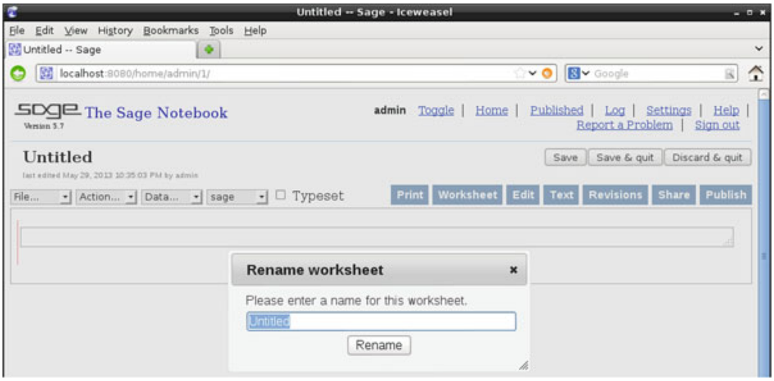


Fig. 2.49 New Worksheet

```
s = singular.eval
s('LIB "surf.lib";')
s("ring rr0 = 0,(x1,x2),dp;")
s("ideal I = x1^3 - x2^2;")
s("plot(I);")\\
```

2.6 Emacs

Knowledge of the text editor *Emacs* can be considered a fundamental skill that will be useful in many ways. *Emacs* has a long history, but it is still popular with professional users. It is unusual in that someone who learned how to use *Emacs* 30 years ago can still use it in the same way now. TEX and UNIX are also classics in the same way. *Emacs* can be hard to learn at first, but once learned, it is an excellent and indispensable tool. Many software systems incorporate aspects of *Emacs*; for example, some of *Emacs* key bindings are used for terminals.

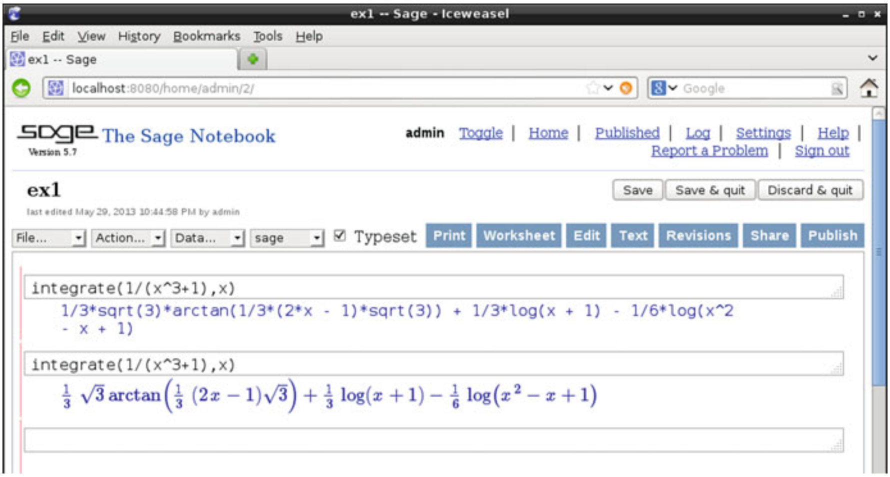


Fig. 2.50 integrate() in *Sage*

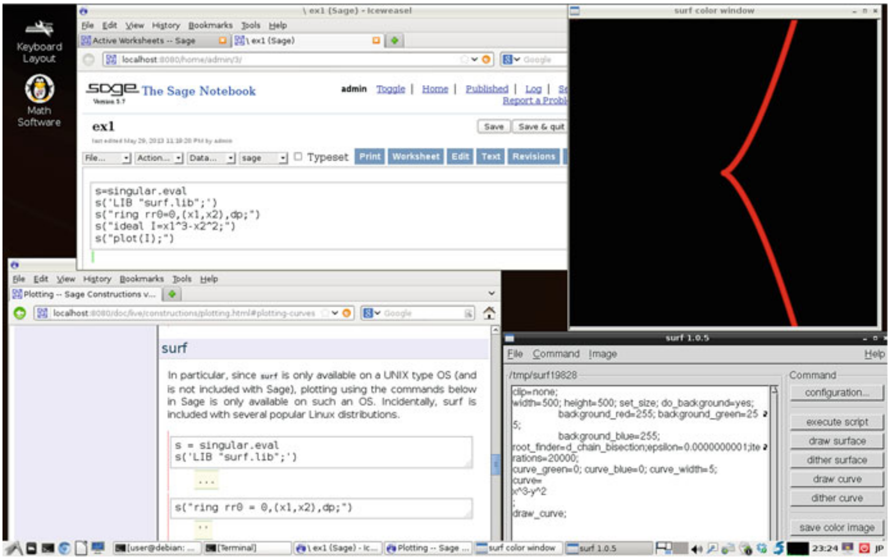


Fig. 2.51 *Singular* and *surf* in *Sage*

In this book, we perform computations using text files that contain commands or scripts of mathematical software. We need a text editor for these files, and *Emacs*, once learned, allows our work to proceed smoothly. For using some mathematical software, such as the programming language Lisp, we can use *Emacs* as an environment.

When we are using computers, inputting characters with a keyboard or drawing with a mouse, we typically are not aware that these manipulations involve reading data from, or writing data to, a buffer on the memory drive.

The official name of *Emacs* is “GNU Emacs”. It was developed by Richard Stallman, who proposed the idea of free software. It was developed and published by the FSF (Free Software Foundation) and is available for free.

2.6.1 Starting Emacs

In this section, we will introduce only a small subset of the many features of *Emacs*. If you are already familiar with *Emacs*, you can skip this section.

If you want to learn *Emacs* thoroughly, there are many specialized books to help you do so. For an introduction, however, open the terminal, move to the directory that contains a tutorial that we made, `~/tutorial`, and execute *Emacs*.

```
user@debian:~$ cd tutorial
user@debian:~/tutorial$ emacs
```

In this example, we call *Emacs* without arguments.

When we execute *Emacs*, a welcome message is displayed. The most important thing is how to quit *Emacs*, which can be done by using the mouse to select “File → Quit” from the menu. It is also possible to quit by typing and entering `C-x C-c`. This is the abbreviation for typing “`x`” and “`c`” while pressing the “`Ctrl`” key.

After booting *Emacs*, press the “`q`” key, and the following will be displayed.

```
;; This buffer is for notes you don't want to save, and for Lisp evaluation.
;; If you want to create a file, visit that file with C-x C-f,
;; then enter the text in that file's own buffer.
```

This is an area called a *buffer*, which is used for short memos and executing Lisp code. It shows that the command for saving a file is `C-x C-f`. If we want to use a mouse, we can select “File → Visit New File…”. If we are familiar with the text editor of Windows or MacOS, we can use *Emacs* by clicking the menu interface. In order to demonstrate the full capability of the original *Emacs*, however, we should use only the keyboard. At first, it is sufficient to use *Emacs* with only the graphical user interface, but it is best to become familiar with other capabilities presented in textbooks and trying them for yourself (Fig. 2.52).[16]

Here we show how to use *Emacs*. We will create a file named “emacs.txt”. First, we need to prepare the buffer for the file; the command for this is `C-x C-f`.

```
Find file: ~/tutorial/
```

[16] *Emacs* always uses the Ctrl key, and so it would be better to change the keyboard layout and set the key to the immediate left of “A” as a Ctrl key.

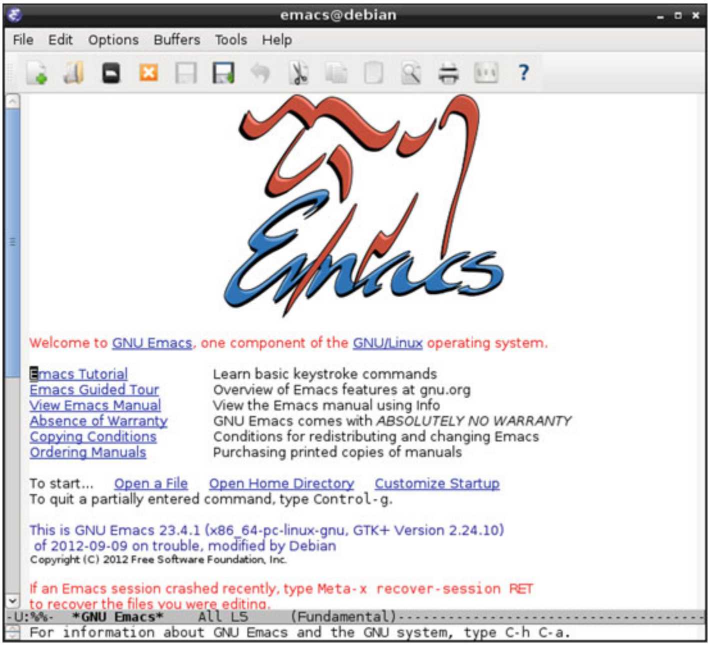


Fig. 2.52 Initial message of *Emacs*

This is called the *echo area*. In the echo area, type the following and press Enter.

```
Find file: ~/tutorial/emacs.txt
```

We have now prepared the buffer for creating the file `~/tutorial/emacs.txt`. We can find `New file` in the echo area, and please input the following.

```
Emacs is a screen editor.
```

After typing the sentence, make a new line by pressing the Enter key. We operate *Emacs* with the “Ctrl” and “Meta” keys. The abbreviation `C-x` means to press the “x” key along with the “Ctrl” key. For example, the command `C-x o` means press “x” with the “Ctrl” key and release the finger from the “Ctrl” key and press “o”. The command `M-x` means press the “x” key along with the “Meta” key. “Meta” keys, however, were found on earlier computers but not on modern PCs; instead, use the “Alt” key. Alternatively, press the “Esc” key, release it, and press the “x” key.

We now want to save the data of the buffer named “emacs.txt” to a file. The command to save a file is `C-x C-s`. When we enter this command, the following message is displayed in the echo area.

```
Wrote /home/user/tutorial/emacs.txt
```

We have now saved the data from the buffer to the directory “tutorial” in our home directory. To quit *Emacs*, input the command `C-x C-c`.

We can check for the created file as follows.

```
user@debian:~/tutorial$ ls emacs.txt
emacs.txt
```

We can find out the size of this file as follows.

```
user@debian:~/tutorial$ ls -l emacs.txt
-rw-r--r-- 1 user user 26 2009-09-14 10:34 emacs.txt
```

We can view the contents of this file as follows.

```
user@debian:~/tutorial$ cat emacs.txt
Emacs is a screen editor.
user@debian:~/tutorial$
```

Finally, using the command `emacs` with the file name as an argument, we can edit this file.

```
user@debian:~/tutorial$ emacs emacs.txt
```

When we open this file with *Emacs*, the contents will be displayed.

2.6.2 Cut and Paste

As an example, we will now edit the buffer and change the word "screen" to "text". When we open the file, the character "E" at the beginning of the line should be blinking. If we input a character, it will be displayed in this location. This blinking part indicates the location of the cursor, which we can move by using the arrow keys. We can also use the Ctrl key to move the cursor: Using `C-f` to move the cursor to the character "n" of the word "screen", remove the word "screen" with the `BS` key, then enter "text". The buffer is now edited, and we will see the following.

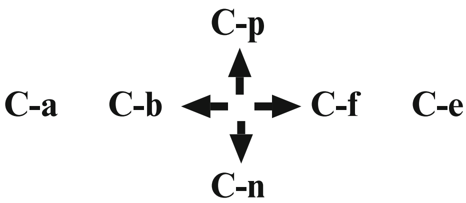
```
Emacs is a text editor.
```

At this point, the cursor is in the position of the character "t" at the end of the word "text".

2.6.3 Editing Multiple Lines

We can edit multiple lines, as follows. Move the cursor to the next line with the command `C-n`. After the word "editor", there will be a control character indicating a new line. After moving the cursor key, we can now edit the next line.

```
Emacs is a text editor.
Emacs is a computer environment.
```



Key binding	function
`C-f`	forward-char
`C-b`	backward-char
`C-p`	previous-line
`C-n`	next-line
`C-a`	beginning-of-line
`C-e`	end-of-line

Fig. 2.53 Cursor moving of *Emacs*

The features of *Emacs* are implemented using the *Emacs* Lisp programming language. The input of characters and moving of the cursor are realized by calling Lisp functions. To explicitly call a Lisp function using its name, input `M-x`, which opens a mini buffer in the echo area. Enter the function name.

```
M-x
```

For example, if we input `M-x forward-char`, the cursor moves to the next character. Just to move the cursor, it is bothersome to input a function name; thus, these functions are usually assigned to shortcut keys. We call this binding the keys. At first, we may use the cursor keys, but it is more efficient to keep our hands on the keyboard. Below is a table of *Emacs* key bindings for moving the cursor (Fig. 2.53).

All of the file operations that were introduced in this section have key bindings with Lisp functions.

Key binding	Function
`C-x C-c`	Save-buffers-kill-emacs
`C-x C-f`	Find-file
`C-x C-s`	Save-buffer
`M-x`	Execute-extended-command

Exercise 2.6.1. Input another sentence.

```
Emacs is a text editor.
Emacs is a computer environment.

At its core is an interpreter for Emacs Lisp,
a dialect of the Lisp programming language
with extensions to support text editing.
```

After completing changes, we can save the buffer to a file. The command is `C-x C-s`.

Exercise 2.6.2. With `C-p`, move the cursor to the third line and insert a new line.

```
Emacs is a text editor.
Emacs is a computer environment.
Emacs is the extensible, customizable editor.
At its core is an interpreter for Emacs Lisp,
a dialect of the Lisp programming language
with extensions to support text editing.
```

We will focus on the mode line, which corresponds to the current line number.

Exercise 2.6.3. Move the cursor, and check the line number.

Not all editing operations are assigned to key bindings. For example, the function for displaying the line number is "what-line"; input `M-x what-line`, and the line number is displayed in the echo area. In order to move to a specified line number, there is a Lisp function "goto-line". When we execute the following command,

```
M-x goto-line
```

we will see

```
Goto line:
```

in the echo area; input the desired number and press Enter to move the cursor to that line.

Exercise 2.6.4. Using the Lisp function "goto-line", move to the fifth line. Now, using the Lisp function "what-line", display the current line number in the echo area.

Exercise 2.6.5. Execute the command `C-x C-c` without saving a file. What is the message in the echo area?

In this situation, the following message may be displayed.

```
Save file /home/user/tutorial/emacs.txt? (y, n, !, ., q, C-r or C-h)
```

This happens when there is an edited buffer which has not yet been saved. If you want to save it, enter `y`; if you do not want to save it, enter `n`. For now, save this file by entering `y`.

2.6.4 Remove Again

Again use *Emacs* to open the text file "emacs.txt".

```
user@debian:~/tutorial$ emacs emacs.txt &
```

The position of the cursor is on the first character of the first line, `E`. To remove the character under the cursor, we can use the *Emacs* command `C-d`. Enter `C-d`, to remove the character `E`.

Exercise 2.6.6. Remove the word "Emacs" with `C-d`.

To remove a single line, use the command `C-k`. This key binding will remove all the text from the position of the cursor to the end of the line. When used with the key binding `C-a`, which moves the cursor to the beginning of the line, we can quickly remove a line.

Key binding	Function
`BS`	Backward-delete-char
`C-d`	Delete-char
`C-k`	Kill-line

What should you do to remove multiple lines?

2.6.5 Point, Mark, and Region

In *Emacs*, the words "point", "mark", and "region" have special meanings.

Concept	Meaning
Point	The position between the previous character and the current position of the cursor
Mark	a Unique point that the user can save to the buffer
Region	The area between the positions of point and mark

For example, here the cursor is located on the indefinite article `a`,

```
Emacs is a computer environment.
Emacs is the extensible, customizable, self-documenting, real-time editor.
```

The point is located between the blank space " " and `a`. We will set the mark at this point; the command for setting the mark is `C-SPC`, where `SPC` means the space bar. We now move the cursor to the character "`t`" of the definite article `the`.

```
Emacs is a computer environment.
Emacs is the extensible, customizable, self-documenting, real-time editor.
```

The region is now as shown here.

```
         a computer environment.
Emacs is
```

In MathLibre, the specified region is displayed with a yellow background. In another environment, the mark and region may be invisible. In that case, input `C-x C-x` twice to find the position of mark. There is another choice, the LISP function "transient-mark-mode", which changes the mode of the region displayed.

When you have verified the region, now try to remove it. The command for removing it is `C-w`. The removed region will be saved to a buffer called the "kill ring". We can now reinsert the region that was saved to the kill ring by using the command `C-y`. There is also a command for duplicating the region, `M-w`. Generally, deleting a region or line `C-w` and `C-k` is called "cut"; duplicating the characters with `M-w` is called "copy", and inserting characters with `C-y` is called "paste". Moving text is called "cut & paste", and duplicating text is called "copy & paste".

By the way, the kill ring is in the shape of a ring. Using `M-y` after the command `C-y` obtains the previous elements in order.

Key binding	Function
`C-w`	Kill-region
`M-w`	Kill-ring-save
`C-y`	Yank
`M-y`	Yank-pop

2.6.6 *Undo, Redo, and Etc.*

If a mistake is made when entering a key or calling a function, it can be canceled with the command `C-g`. This is a very important and useful command.

In the following table, we summarize the *Emacs* Lisp functions which will be necessary in addition to the features described in this section. *Emacs* can be operated with a mouse, but it is faster and easier if only the keyboard is used.

Key binding	Function
`C-g`	Quit
`C-x i`	Insert-buffer
`C-x C-w`	Write-file
`C-s`	isearch-forward
`C-r`	isearch-backward
`M-%`	Query-replace
`C-x o`	Other-window
`C-x u`	Advertised-undo
`C-_`	Undo

When the command `M-x help-with-tutorial` is entered, a basic introduction of *Emacs* will be displayed. You should read it.

2.6.7 Command and Shell

We introduced *Emacs* as a text editor. At this point, we will introduce the extended features that use the Lisp language and show how it can be used as a computer environment.

In Sect. 2.3, we entered various commands. We can also use a variety of commands with *Emacs*. If we enter `M-!`, "Shell command:" will be displayed in the echo area. We can now execute a command. For example, after executing `M-!` and entering `ls`, a list of the files in the current working directory will be displayed.

If we want to input commands exclusively, we can execute the *shell* in *Emacs*. The command to change to the *shell* mode is `M-x shell`. We will then see the prompt of the *shell* and can use it in a way similar to the terminal. For example, we can move the cursor and copy and paste regions.

In addition, MathLibre also contains a file manager (`M-x dired`), the Tower of Hanoi (`M-x hanoi`), Goban (`M-x gomoku`), and Tetris (`M-x tetris`). All of these are implemented using the *Emacs* Lisp language.

2.6.8 Math Software Environment

Some mathematical software systems can be used with *Emacs*. We summarize the commands in the following table. When we use software in *Emacs*, it is very similar to using the *shell* on *Emacs*. Executing mathematical software on the terminal is easy and simple, but using a system on *Emacs* may be convenient because of the various features of *Emacs*. In particular, the computer algebra system *Maxima* contains a Lisp package named *imaxima*; this package allows us to elegantly typeset our results, Fig. 2.54.

Maxima	`M-x imaxima`
Macaulay2	`M-x M2`
Singular	`M-x singular`
R	`M-x R`

2.7 Other Ways of Booting MathLibre

When we boot MathLibre from a DVD, some think it is too slow and noisy. We would like to recommend the use of a virtual machine, which is a software-implemented version of hardware that runs at the applications layer of the operating system. A virtual machine can emulate the computer architecture of Windows, MacOS X, or Linux.

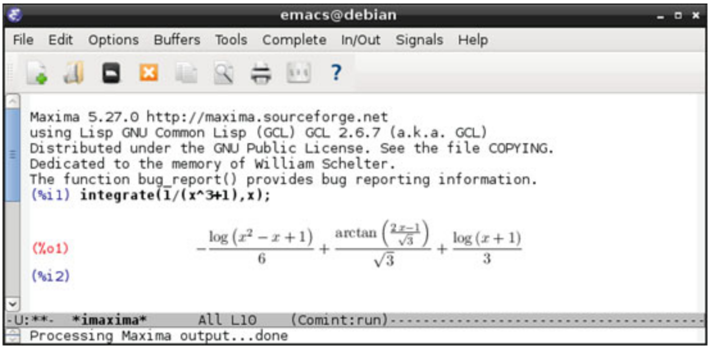


Fig. 2.54 M-x imaxima

Here are two the virtual machines which can be freely downloaded: *VMware Player* of VMware, Inc. and *VirtualBox* of the Oracle Corporation. *VMware Player* is an application for Windows and Linux; MacOS X users can buy a commercial product, *VMware Fusion*. *VirtualBox* is an open-source product that can be used on Windows, MacOS X, and Linux.

Currently, there are some virtual machine image files of MathLibre for *VMware Player* that can be downloaded from http://www.math.kobe-u.ac.jp/vmkm/. These commands shown in the image files support a persistent home directory, which can be used for daily work. In September 2009, we hosted a JST CREST Gröbner school at Kobe University; it was for graduate students who wanted to learn about Gröbner bases and mathematical software. We used a virtual machine as the standard computer environment and can thus recommend it as an everyday environment for research.

2.7.1 Various Virtual Machines

Here are some virtual machines for related project. KNOPPIX/Math was a project for archiving mathematical software and free documents in KNOPPIX, another Live Linux. MathLibre is a direct descendant of KNOPPIX/Math. All of them are for *VMware Player*. If you use Windows, you will have to install the VMware Player; for MacOS, install VMware Fusion. These virtual machines are compressed and must be extracted.

`VMware/KNOPPIX/Math 2008` This is a virtual machine for "knxm 2008-kobe.iso". It is the latest version of KNOPPIX/Math with a KDE desktop and a multi-lingual environment.

`VMware/Knoppix/Math 2010(en)` This is a virtual machine for KNOPPIX/Math 2010. You will also need the ISO image of "knoppix_v6.2.1-math-dvd-icms2010-20100730-en.iso"

`Small VM/KM` These are the small virtual machines lenny-ox and etch-ox. These are old Debian distributions with the OpenXM package. They contain *Risa/Asir* as a sample application.

`MathLibre 2013` This is the most recent virtual machine for MathLibre 2013. You will also need to download the ISO image of MathLibre 2013.

2.7.2 Making a USB-Bootable MathLibre

This is a third way to boot MathLibre. We are currently experimenting with implementing a shell script that makes MathLibre bootable from a USB device. A USB flash drive is a very convenient and economic device. With a USB-bootable version, when we want to create a mathematical software environment for our students, we can copy it to bootable devices and redistribute them. It takes about 20 min to copy the operating system and applications onto a USB device.

1. Boot with MathLibre DVD.
2. Connect a USB flash drive that has over 8 GB of memory.
3. Execute the following command.

```
user@debian:~$ sudo mkusbmath
```

4. View the list of USB devices.

```
-----------------------------------------------------------
 mkusbmath: shell script for making USB bootable MathLibre
-----------------------------------------------------------

Please select the target device from the following list:

/dev/sdb usb-Generic-_SD_MMC_058F63626420-0:0
/dev/sdc usb-Generic-_Compact_Flash_058F63626420-0:1
/dev/sdd usb-Generic-_SM_xD_Picture_058F63626420-0:2
/dev/sde usb-Generic-_MS_MS-Pro_058F63626420-0:3
/dev/sdf usb-ELECOM_MF-LSU2_eb972e623b2081-0:0

Please input your target device (ex. /dev/sdc) :
```

5. Chose /dev/sdf, input “`/dev/sdf`”, and press Enter.

```
/dev/sdf
Unmounting the mounted partitions.

We've selected the device: /dev/sdf

By the following operation,
all files in /dev/sdf will be !!!REMOVED!!!.

It takes over 20 minutes for making,
would you start this operation? (y/n)
```

6. If you input y, then all the files in the flash drive will be removed. If that is OK, please input "y" and press Enter.

```
y
You're  copying OS image to /dev/sdf
  1.3GB at  112.1MB/s  eta: 0:00:22 34 [======                   ]
```

7. The process will take more than 15 min. You will see a progress bar; it may take a few minutes after reaching 100 %.
8. After the copying is completed, a persistent home directory is automatically created.

```
We've finished copying OS image.
We're making persistent volume.
```

9. The following message is displayed when the procedure is completed.

```
We've finished making persistent volume.
You've got an USB bootable MathLibre.
```

MathLibre was developed using the Debian Live system.[17] Debian GNU/Linux is a Linux distribution, and it uses the APT system to manage packages. We can install additional Debian packages with the command `apt-get`. To install additional packages, we have to update the resource database. To do this, we have to change to administration mode by using the command `sudo`.

```
user@debian:~$ sudo apt-get update
```

The command sudo executes a command as another user. For example, if we want to add another general purpose computer algebra system, "`axiom`", we input the following command.

```
user@debian:~$ sudo apt-get install axiom
```

Note that all packages that are required by the specified package will also be retrieved and installed.

2.7.3 How to Install MathLibre to an Internal Hard Disk

Beginning with MathLibre 2013, the installation of MathLibre to an internal hard disk is supported. The install menu can be found while booting, Fig. 2.55; when the "Install" or "Graphical install" menus are selected, we can install MathLibre as well as ordinary Debian distributions. There is an manual for installing the Debian Project.[18]

[17] http://live.debian.net/.

[18] http://www.debian.org/releases/stable/installmanual.

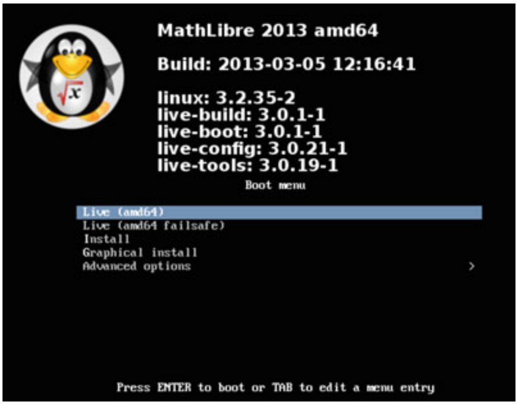


Fig. 2.55 MathLibre boot menu

References

1. D. Cox, J. Little, D. O'Shea, *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra* (Springer, Berlin, 2006)
2. M. Gancarz, *The UNIX Philosophy* (Digital Press, Bedford, 1994)
3. B.W. Kernighan, R. Pike, *The Practice of Programming* (Addison-Wesley, Reading, 1999)
4. MathLibre Project, http://www.mathlibre.org/
5. J. Moses, Macsyma: A Personal History (May, 2008), http://esd.mit.edu/Faculty_Pages/moses/Macsyma.pdf
6. Risa/Asir Drill Book, http://www.math.kobe-u.ac.jp/Asir/asir.html

Springer

http://www.springer.com/978-4-431-54573-6

Gröbner Bases
Statistics and Software Systems
Hibi, T. (Ed.)
2013, XV, 474 p. 123 illus., Hardcover
ISBN: 978-4-431-54573-6