Introduction to Computing
Lecture notes and computer exercises

Karin Carling

Göteborg 2001

Contents

Chapter 1

Basic Unix

First you will learn some basic commands for the Unix-system, the basics about the file-system, and how to make text-files with the editor Emacs. This knowledge is essential if you are going to use the computers at Djungeldata.

1.1 First meeting with the computer

Sit down at a workstation. It is usually turned on. If the screen is black, move the mouse and wait for a little while. (The screen goes black to save energy when no-one have been using the computer for a while.) You will see a login-prompt at which you have to write your user-name and you will be asked for your password. When the login process is finished you will have a couple of windows on the screen. Move the pointer into one of the windows. This window will be activated and you can type Unix-commands here.

1.1.1 Commands

1. Write some simple commands, e.g. `date`, `cal` and `who`.

2. Misspell on purpose and correct it. You can move back and forth on the command line with the arrow-keys, $\leftarrow$ and $\rightarrow$, and remove characters in front of the marker with `Backspace`. It is the key above the return-key with an arrow pointing to the left. You can also use `Control-d` (the Control-key is held down simultaneously with the `d`-key, short: `C-d`) if you want to remove the character which the marker covers.

   Tab-expansion is also good to know. The command interpreter may fill-in your commands and file names. Write `xre` and press the `Tab`-key, and you will get `xrefresh`. This will work if there is an unique choice to make. If you write `ma` and try tabular-expansion, you will get a beep as the answer. Now write `C-d` and you will get a list of the commands beginning with `ma`.

3. Give the command `man` for some program, e.g. `who`. (New page = space, end listing = q).

4. Go through your previously executed commands with the arrow-keys $\uparrow$ and $\downarrow$. Try the command `history`.

1.1.2 Simple file-system exercise

Write the commands you use, and the resulting output.

1. Find out in which directory you are.
2. Move two steps upwards in the library tree, in one command. List the files in this directory.
3. Move to your home directory.

1.2 Basic Emacs

There are a lot of different programs for text editing. Here you will learn the basics of an editor called *Emacs*. Start Emacs by writing `emacs &` in some window*. End Emacs by choosing `Exit Emacs` under the `File`-menu(short: `File:Exit Emacs`) or with `C-x C-c`.

Try editing some text. Save it to a file `File:Save Buffer As` and give it a name.

To learn more, try the Emacs tutorial (`Help:Emacs Tutorial`). I recommend that you take some time to work it through. There are a lot of useful tips and tricks in the tutorial. Also, there is a short list of key-bindings that you might find useful in appendix A.2.

Exit Emacs.

1.2.1 Make a file

You will now make a file in Emacs. Open Emacs again and write the following text:

```
Mike 345
Tom 968
Kim 23
Mary 136
Evelyn 485
```

Save this text to a file as `testfile`. Move the pointer to an xterm-window and make sure that you are in your home directory. Make sure that you have the `testfile` in you present working directory and that the contents is correct by printing the file to the window with `cat`. Move the mouse to the Emacs-window and change the file contents by adding the row "Jane 295". Save the file and exit Emacs.

1.3 More exercises

Write the commands you use to do the following tasks:

1. Copy the contents of `testfile` to a new file `copyfile` and control the existence of the new file with `ls`.
2. Change the name on `copyfile` to `indata` and remove the file `testfile`.
3. Check, by using `grep`, if "Tom" is in the file `indata`.

*The & will put the program in the background, and you will not block the command line for new commands. Very convenient.

1.4 How to use the manual

In this part of the exercise you will learn how find information about a command by the help of the manuals, which are available through the command `man`.

Write the commands you use to solve the following tasks:

1. Find out on which day of the week you are born with `cal`.
2. Which day of the week was September 3, 1752?
3. Count the number of characters, words and lines in the file `indata` with `wc`.
4. How do you write to make `wc` count words only?
5. What is the action of the flag `-a` to the command `ls`?
6. What can you use the command `apropos` for?
7. Sort the file `indata` alphabetically, with the command `sort`.
8. Sort the file `indata` numerically by the second column.

1.5 Redirect and pipelining

Most of the commands write their output to the screen. In Unix it is easy to *redirect* the output to a file with `>` followed by a filename.

```
boromir> ls
file1 newfile other.file my.file
boromir> ls > listres
boromir>
```

Commands that normally read their input from the terminal can be made to read this data from a file with `<`. If we would like to count the number of words, i.e. the number of files in the present directory, in the file `listres` we created above we can do the following.

```
boromir> wc -w < listres
  4
```

It is not necessary to make a file every time you would like to count the number of files in a directory, you can connect the commands with a pipe (|). This will cause the output from the first command to be piped to the other command.

```
boromir> ls | wc -w
```

The redirects shown above is only the basic ones. If you are curious of more information about this read the manual pages of the command shell, i.e. `man csh`.

And finally a little exercise (write the commands you use):

1. Sort the file `indata` in the inverse alphabetical order and save the result in the file `atadni` with a redirect.

1.6 File quota

Every file you have will need disk-space for storage. Do the command `ls -l`. How much space do the file `indata` take? Try the command `quota` to find out how much disk space you are allowed to use. Assume that you get the following answer:

```
Disk quotas for f92kaan (uid 7580):
Filesystem     usage  quota  limit    timeleft  files  quota  limit    timeleft
/users/dd/mp97
                   5  50000  75000                  6      0      0
```

This output shows that you use 5 kb disk-space of the allowed usage of 50000 kb, that is 50 Mb, and during a short period of time you may use up to 75 Mb of disk-space. You have 6 files stored and there are no limits to the number of files that you are allowed to have on the file-system.

How much disk-space are you using? If you get a message that you have used more disk-space than your quota allows you should clean up immediately on you account, that is delete files.

1.7 Security

The world is full of wicked people who enjoy to steal or destroy other peoples work, including yours, and they are at Chalmers too, actually. Therefore, it is of uttermost importance that you make everything in your power to prevent unauthorized access to your account.

1.7.1 Lock your account!

If you leave the computer for a short time, only a few minutes, someone might remove all your files, steal you printer-quota and send offensive e-mails in your name! To avoid this, lock the workstation every time you leave it unnoticed.

To avoid the misuse of `xlock` there is a time limit (usually about 20 minutes) on the command. When the time is up, anyone may log you out.

1.7.2 Change password!

To stop unauthorized use of your account it is important that you keep your password a secret. You should change the password you where given when you got your account with the command `nispasswd`. You are not allowed to have a password that can be found in dictionaries or otherwise is too "easy".

If you forget your password you have to ask the system administrators (fop) for a new one. System administrators usually dislike people who keep forgetting their passwords. I suggest that you remember your password.

1.8 Finally

The command-list, see appendix A.1 contains a lot of useful commands and programs. The following commands are very useful, I suggest that you learn them by heart: *cat, cd, cp, emacs, grep, kill, lpq, lpr, lprm, ls, man, mkdir, more, mv, nispasswd, pine, ps, printerquota, pwd, quota, rm, rmdir, top, wc, xlock.*

1.9 Log off

When you are finished working at the workstation you have to quit your programs and log off. *Always quit programs by their exit/quit command!* (Be extra careful to do this with Netscape and Matlab!) When you have done this you click on the button marked “logout”. Do **not** turn off the power! The screen will go into an energy-saving mode automatically.

Make sure that you clean up the desk before you leave the computer room!

1.10 How to learn more

There are a lot of good information available on the web, e.g. at the Djungeldata homepage. Try these links for more information about Unix:
`http://www.dd.chalmers.se/UNIXhelp/`
The Faq-O-Matic:
`http://www.dd.chalmers.se/fom-serv/cache/1.html`

If you would like to pick up a book to learn more about Unix there are a lot of books available at the main library at Chalmers. Here are some examples of books which I have found useful:

An Introduction to the Unix Operating System C. Duffy. A bit old but short introduction.

The Unix Operating system K. Christian and S. Richter. The first part of the book covers basic Unix knowledge, and the second part covers more advanced Unix skills.

Unix Clearly Explained R.L. Petersen. This book is similar to the previous one, but a bit less advanced.

Chapter 2

Introduction to LaTeX

There are a lot of programs for writing reports and other documents. Some of them are similar to Word, e.g. StarOffice, some are more enhanced, e.g. FrameMaker, and then there is LaTeX. This part will teach you the basics of LaTeX since it is a powerful typesetting program which can be found, or installed for free, on almost any computer system. To be able to use LaTeX, you also need to know how to use a text editor, e.g. Emacs.

2.1 Getting started with LaTeX

LaTeX is a *mark-up language*. The document you are working on will not appear directly on the screen until it is typeset and opened in a viewer. You will be working with the text in a *source file* which will include *mark-up commands* that indicates how to typeset the document. This might seem complicated but it will give you many advantages when writing equations and documents with cross-references, and LaTeX is the document format that many physics journals and conferences prefer when you want to submit your contributions.

The basic structure of a LaTeX source file is shown in Fig. 2.1. The *preamble* specifies the document *class* and then the *packages* used by the article. Also, you can include specialized commands that will be used throughout the document and other global definitions.

The *body* of the document is enclosed by

```
\begin{document}
...
\end{document}
```

which marks the enclosed text as the part of the document that is to be typeset according to the "rules" of the preamble, and the mark-up commands in the text of the body.

In the body you have the *abstract*, the main text which is divided into *sections*, and in the end a list of literature references, the *bibliography*.

To get a typset document from the source file (`ltest.tex`) you will have to run the following commands:

```
latex ltest
xdvi ltest.dvi &
```

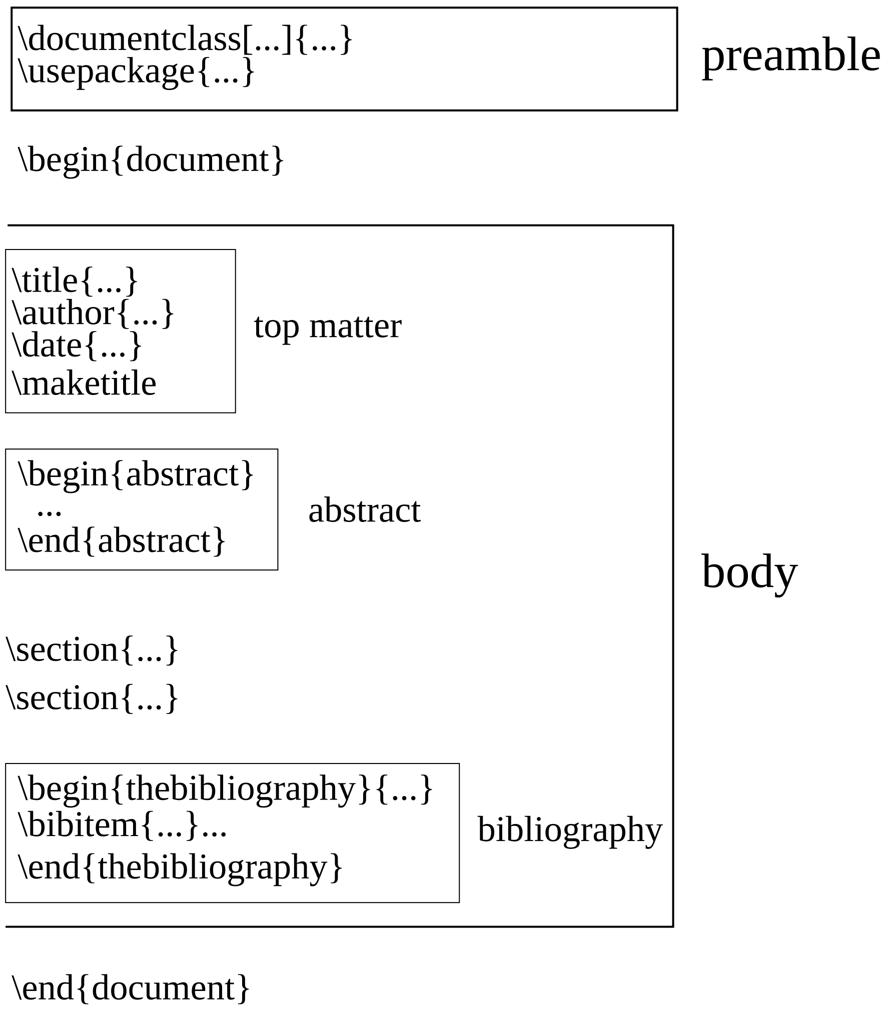

Figure 2.1: The basic structure of a source file

The formatted document will appear in a new window on the screen. If you want to print the document you have to apply one more command to the file:

```
dvips ltest.dvi
```

This will generate a file `ltest.ps` which can be viewed (`ghostview`) or printed (`lpr`).

2.1.1 Typing text

Text are typed into any LaTeX document with the following keys

```
a-z   A-Z   0-9
+ = * / ( ) [ ]
```

and the following punctuation marks

```
, ; . ? ! : ‘ ’ -
```

and the space-bar, the tab key (which gives the same effect as the space-bar in the typeset document), and the Return key. There are also thirteen keys that are mostly used in LaTeX instructions:

```
# $ % & ˜ _ ˆ \ { } @ " |
```

If you want to typset these caracters there are commands available, see App. A.3.3.

The other (swedish) keys on your keyboard cannot be used, unless you include a package that can handle this input. At Djungeldata your can include the `inputenc` package.

To give you an idea of how the source file might look like, here is a short example of how to write some text in LaTeX:

```
\documentclass[a4paper, 12pt]{article}
% This is a comment which will be ignored.
\begin{document}
\section{First section}
\label{thefirst}
Here you can write the text you like.  The number of
spaces between the words     will be ignored when the
 text is typset
as well as
any single linebreaks.

A new paragraph is defined by two or more linebreaks,
that is one or more empty lines.


Sometime you might want to \emph{emphasize} a word.

\end{document}
```

The source file begins with

```
\documentclass[a4paper, 12pt]{article}
```

which specifies the class of the document (`article`) and includes the optional arguments `a4paper` and `12pt` which states that this document should be typset for the paper size a4 and with the font size 12 points.

The second line is a comment. When LaTeX encounters the %-character the rest of the line in the source file will be ignored. This can be conveinient if you want to add some notes to your source file.

The text of the document is typed within the `document` environment. An environment is the region between any pair of

```
\begin{...}    \end{...}
```

where the enviroment type is given within the braces.

The text is sectioned with the `\section` command which can be accompanied by a `\label` command that ties a short name to the section heading to make it easy to refer to this section later in the document, see Section 2.1.4. There are several levels of sectioning available. The number of levels depend on the class of the document.

As can be seen from the typeset document, LaTeX notices that there is a space or tab in the file but it ignores how many, and puts only one space in the typset document. The same is true for blank lines, which will cause LaTeX to make a new paragraph. Usually the spacings will be typeset correctly, but once in a while you might run in to trouble. You might want a space between two words but you do not want a linebreak between the words. Then you can "tie" the words together with the ˜(tilde), which will give a nonbreakable space.

Another problem you might encouter is that the hyphenation is incorrect. Usually, english words will be hyphenated correctly, but there might be names of technical terms that are difficult to handle. To help with the hyphenation, you can insert one or more *optional hyphen* commands (`\-`) to show LaTeX where the word might be hyphenated. If the word occurs many times in the document, you can use a "global solution", by putting the word with optional hyphens in the preamble instruction

```
\hyphenation{in\-struction}
```

2.1.2 Typing math

Most problems in physics are connected with equations and formulas. LaTeX is an excellent tool for typesetting equations and mathematical notations. Equations in LaTeX can be *inline* or *displayed* environments. The following example of a source file and the result can give you an idea of how the math environments work.

```
In first year calculus, we define intervals such as
$(u, v)$ and $(u,\infty)$. Such an interval is a
\emph{neighborhood} of $a$ if $a$ is in the interval.
Students should realize that $\infty$ is only a symbol,
not a number. This is important since we soon introduce
concepts such as $\lim_{x \to \infty} f(x)$.

When we introduce the derivative
\begin{equation}
  \label{eq:derivative}
  \lim_{x \to a} \frac{f(x) - f(a)}{x - a},
\end{equation}
we assume that the function is defined and continuous
in a neighborhood of $a$.
```

The typeset document will look something like this:

In first year calculus, we define intervals such as $(u, v)$and (u, ∞). Such an interval is a *neighborhood* of a if a is in the interval. Students should realize that ∞ is only a symbol, not a number. This is important since we soon introduce concepts such as $\lim_{x\to\infty} f(x)$.

When we introduce the derivative

$$\lim_{x \to a} \frac{f(x) - f(a)}{x - a}, \tag{2.1}$$

we assume that the function is defined and continuous in a neighborhood of a.

The inline environment open and close with $, and the displayed environment is contained within

```
\begin{equation}
...
\end{equation}
```

The `equation` environment includes the possibility to label the equation for easy cross-referencing, see Section 2.1.4, with the `\label{name}` command, and the equation will be numbered.

There are many commands within the math environments to give the components of an equation. LaTeX includes almost any mathematical symbol you might like to use, and there are a number of packages that give you access to even more symbols. A list of the most common components is found in App. A.3.4.

2.1.3 Graphics and tables

Usually, the results from an experiment or calculation is presented in a graph. To illustrate you text by importing graphics into your document is possible in LaTeX. The file formats that can be imported to a LaTeX document is Encapsulated or plain PostScript. Usually, these files have the suffices `.eps` or `.ps`.

To be able to import graphics into the document you must include the `graphics` package in the preamble. This package includes the commands

```
\includegraphics
\scalebox
\rotatebox
```

which might come in handy when you are importing your graphics.

The graphics you like to import are placed within the `figure` environment. Below is an example of the command series used to put a centered figure (`my-fig.eps`) into a document.

```
\begin{figure}[htbp]
  \begin{center}
    \includegraphics{my-fig}
    \caption{The caption of the figure is written here.}
    \label{fig:myfigure}
  \end{center}
\end{figure}
```

The `includegraphics` command assumes that the file suffix is `.eps`. If there is no such file it looks for a file with suffix `.ps`. The figure can be labelled for easy cross-referencing, and you can include a figure caption with explanatory text.

Sometimes your data is better shown as a table. In LaTeX you will construct a table with the `tabular` and `table` environments. The following text in the source will produce the table as shown below.

```
  \begin{table}[h]
    \begin{center}
      \begin{tabular}{| l | l | r|}
        \hline
        First & Second & Third \\
        \hline
        34 & 678 & 2020\\
        \hline
      \end{tabular}
      \caption{This is the caption of the table}
      \label{tab:name}
    \end{center}
  \end{table}
```

This table will be the result

First	Second	Third
34	678	2020

Table 2.1: This is the caption of the table

2.1.4 References and bibliography

When a document begins to contain more than five to ten pages, figures, equations, or bibliographic references it becomes difficult to keep the cross-references in order. LaTeX includes commands that makes life much more easy when it comes to cross-referencing. You have already encountered the `\label` command that will give you a name for every figure, table, or equation in your document. These names are used with the `\ref{name}` command to make a reference to the figure, table, or equation called `name`. Also, you can make references to the different sections of your document.

To cite a book or article you need to create a list of the bibliographic references you use in your document. This list is created in the `thebibliography` environment, and each item in this list is a `bibitem`. The `bibitem` has a label which is used for referencing the book, or article, with the `\cite{name}` command.

2.2 How to learn more about LaTeX

You can get almost anything out of LaTeX. The material you get in this course is all written in LaTeX. You can make a reference list, include pictures, write complicated equations and include footnotes. Everything, which is not simple text, is "marked" with format marks to tell the typesetting program, i.e. LaTeX, how to do format the text. In App. A.3 there are some examples of environments and commands which are used a lot for typesetting with LaTeX.

Try to write a small report, a couple of pages, with at least an equation, a table, a book reference, a footnote and a picture. To learn more on how to include all these items look at the Djungeldata-LaTeX home page:
`http://www.dd.chalmers.se/latex/index_e.html`.
There you will find nice examples of reports and other documents you might want to write.

To get a more or less complete list of commands to LaTeX you can go to:
`http://mitra.phys.uit.no/edb/latex/index.html`.

There are books about LaTeX available at the main library at Chalmers and here are some of the books I have found useful:

First Steps in LATEX G. Grätzer. If you read and do the exercises of this book you have achieved the skills you need to make an article. Quite short.

A Guide to LATEX H. Kopka and P.W. Daly. A more comprehensive book.

LATEX – a document preparation system L. Lamport. *The* book about LATEX. If you should buy one book, buy this one.

The LATEX companion Goossens, Mittelbach and Samarin. This is the second mostly bought book about LATEX.

Chapter 3

Computing and programming with MATLAB

This part will get you started with MATLAB and give you the basic knowledge for using the program. If you are going to pass the Computational Physics course in the spring it is important that you continue to practice and improve on your MATLAB skills.

3.1 How to get started

3.1.1 Start and stop

To start MATLAB you type `matlab` at a command-prompt in an available window, or click on the MATLAB-icon. Whenever you want to quit MATLAB you type `quit` at the MATLAB prompt. Do not forget to quit MATLAB before you logout!

There are some commands you can try immediately, e.g. `tour` and `demo`. The last one is very useful for getting started with MATLAB. To get help you should learn how to use the commands `help`, `helpwin` and `helpdesk`.

If your computation is running out of hand, you can force it to stop with `ctrl-c`, which will send a stop signal to the process and you will get the MATLAB-prompt back.

3.1.2 Basic calculations

You can use MATLAB as an ordinary calculator. You can try to compute the sum $5011 + 13$. If you want to save the result of a computation you need to assign it to a variable: (In the examples I will only write the MATLAB commands and, on some occasions, the output.)

```
x=14
y=3*x
```

The usual mathematical functions are available within MATLAB, e.g. `sin`. Parentheses are used as usual. If you put a semicolon at the end of a command the computation will be performed as usual but there will be no output on the screen. This is convenient when you are working with large arrays of data.

If you are curious of the variables saved by MATLAB, check them with the commands `who` and `whos`. The last command will give more details about the variables stored. To clear all variables you type `clear`, which can be used to clear one variable at the time too. (How do you do that?)

Variables in MATLAB are vectors and matrices. Enter this and comment on the output.

```
vcol=[1;2;3;4], vrow=[1 2 3 4]
A=[1 2 3 4;5 6 7 8;9 10 11 12]
sqrt(vcol),sqrt(A)
```

If you need a long vector you do not want to enter every number by hand. The colon-operator is a tool for the construction of vectors:

```
vector=0:8
vector2=0:0.5:2
```

You should also learn how to use `linspace` and `logspace`.

You can do operations element-wise on vectors and matrices.

```
values=2.^vector
```

If you have defined some variables you can use those to create new variables.

```
table=[vector;vector.^2;vector.^3]
```

3.2 Simple graphics

To get a graph you choose which type of graph you would like to have, and learn what type of data you need to give the command. One of the simplest graphs you get with `plot`. Try to plot the powers of two you have made previously.

```
plot(vector,values)
```

To get a smoother curve you use more values. Try to make a smooth plot of the powers of 2. Read the help-file and learn how to get different colors and line-types in your graph.

It is also possible to do 3D-graphics.

```
vector2=0:0.5:8;
[X,Y]=meshgrid(vector2);
mesh(X,Y,2.^X+2.^Y);
```

What happens if you change `mesh` to `surf`?

When you make statistics the command `hist` is handy. You can create random numbers with `rand` and check how random they are.

```
random=rand(1,7)
hist(random);
```

3.2.1 Save figure to file

When you have plotted an illustrative figure you might want to put it into your report. To do this you have to save the picture to a file and import the file into the document. There are many formats for pictures but the one recommended in this course is Encapsulated PostScript (eps), which is easy to import into a LaTeX-document.

To save a MATLAB figure to a eps-file you can write the following command:

```
print -depsc -epsi filename
```

or use the menus of the figure window. This will give you a file `filename.eps` in the working directory. This file can be imported into your document. (If you want to learn more about the flags given to the `print`-command try the MATLAB-helpdesk.)

3.3 Systems of linear equations and eigenvalues

When you are doing computational physics you need to know your linear algebra. MATLAB is based on matrices as data format, and there are many built-in tools for extracting information from a matrix and solve a wide range of problems.

Begin with these matrices:

$$\mathbf{A} = \begin{pmatrix} 3 & 4 & 5 \\ 5 & 2 & 2 \\ 1 & 2 & 3 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 1 & 1 \end{pmatrix} \tag{3.1}$$

Enter them into MATLAB. You can multiply the matrices with `B*A`. What happens if you type `A*B`? Why? Try some other matrix commands on these matrices, e.g. `det`, `trace`, `null`, `orth` and `inv`. Try to understand what these commands do.

To solve a system of linear equations e.g.

$$\left\{ \begin{array}{rrrr} 3x_1+ & 4x_2+ & 5x_3 = & 25 \\ 5x_1+ & 2x_2+ & 2x_3 = & 18 \\ x_1+ & 2x_2+ & 3x_3 = & 13 \end{array} \right. \tag{3.2}$$

This system of equations can be written as $\mathbf{A} * \mathbf{x} = \mathbf{b}$ which is solved with

```
  b=[25;18;13];
  x=A\b
```

Check the solution by solving the system of equations by hand.

In MATLAB the `A*B` will give you the matrix multiplication. If you would like to do element-wise multiplication you have to use the operator `.*`. There are other operators and functions that are available as matrix or element operations.

3.4 Simple programming

A program is a list of commands that are to be executed, often controlled by condition and repetition clauses, to solve a certain problem. In MATLAB you write you programs to files with the suffix `.m`.

A MATLAB program can be a routine or a function. If you want to write a function the first line of the file have to begin with `function`, e.g.

```
function y = div(n,d)
```

To avoid problems, do not give your functions the same name as a built-in MATLAB function, and give the source file the same name as the function, e.g. `div.m` for the function `div`.

On the home page of the course there is a program available for you to read and run. Try to understand every command in this program and what they do before you run to see the result.

Try to make some MATLAB programs on you own. You might have some problems from one of the other courses you are taking that you might try and solve with MATLAB.

3.5 How to learn more

To learn more about MATLAB try the MATLAB-helpdesk, available on the web, which includes a tutorial and other resources.

There are quite a large number of books available on MATLAB, for example:

The MATLAB 5 Handbook Pärt-Enander, E et al. (Addison Wesley Longman, 1999 (ISBN 0-201-398451). This is a nice book which you probably will find useful no matter of your previous knowledge of MATLAB. If you are going to buy a book, I recommend this one.

Introduction to MATLAB for engineers and scientists D.M. Etter. A short book for the beginner that includes several exercises. If you need ideas and practice, borrow this book from the library and use it.

Appendix A

Short guide to basic commands

This document contains some lists with the most common commands used on the Djungeldata computer system, and in some of the programs that you will use during your education at the Physics Department. It is a good idea to check that you are familiar with these commands. If not, learn more about them through the manual and help pages.

A.1 UNIX commands

To learn more about a UNIX command, what arguments it takes and which flags that can be used, check the manual page with `man` *command*. For more information about UNIX check this web page: `http://www.dd.chalmers.se/UNIXhelp/`. There is also a lot of other useful information on the Djungeldata web pages.

man *command* Show the manual page for the *command*. I strongly suggest that you learn how to use this command and how to read the manual pages.

apropos *keyword* Locate commands by keyword lookup.

nispasswd Use this to change your password.

xlock Lock the workstation for a short period of time.

pwd Print the path to the present working directory, i.e. the directory in which you currently are positioned.

ls List the files and directories in the current directory. Some optional flags:

- **-l** Long listing, gives information about size, owner, date, and status of files and directories.
- **-a** Show all files and directories, including the hidden ones (.*name*).
- **-F** Shows markers at the file and directory names, e.g. a directory is marked with a trailing slash(/).

mkdir *name* Create a new directory called *name* in the current directory

rmdir *name* Remove the directory called *name*. (This requires that the directory is empty.)

cd *name* Change directory. If no directory name is give you will end up in your home directory.

mv *file1 file2* Move, actually rename, *file1* to *file2*.

mv *file(s) dir* Move the *file(s)* to directory *dir*.

cp *file1 file2* Copy *file1* to *file2*.

chmod *flags file* Change the status (mode) of the *file* or directory. Use this to protect you files. Some flags:

- **u+x** Gives the user (you) the right to execute the file.
- **og-rwx** Removes all rights for other users to read, write or execute your file (directory).

ps List the process numbers of the processes that are running on the computer.

top Allows you to check the load of the computer. Can be used to identify and kill a looping process.

kill *pid* This command kills the process which you tell it to kill. Be careful when using this.

quota -v Show how much of your quota of disk-space that you are using.

printerquota Show how many pages you can print from your account. If you need more pages you can by extra printer quota from F-styret.

lpr *file(s)* Print the file(s) to the default printer.

lpq Show the default printer queue.

lprm *jobid* Remove your printer job from the printer queue.

more *file* Show file contents on the screen one page at the time.

grep *pattern file* Find the pattern in the file.

wc *file* Count the number of lines, words and characters in the file.

sort *flags file* Sort the contents of the file (the flags are optional). You can sort in a lot of ways if you learn how to use the flags.

A.2 Emacs

This is a short list of key bindings that is useful when you are working with emacs. If there is a dash (-) you should hold these keys down at the same time, and key bindings should be done in sequence. The **C-x** is short for holding down the Control-key and x-key at the same time, and **M-x** is the Meta-key and the x-key at the same time. Usually, the Meta-key is a rhombi-marked key.

C-g Interrupt previous command. Use this if emacs is behaving strangely.

C-x C-c Quit emacs.

C-x C-f Find file, i.e. read a file into emacs for editing.

C-x i Insert file into the active buffer.

C-x C-s Save buffer to file.

C-x C-w Save buffer to a new file-name and make that file the active buffer.

C-x b Change to another buffer.

C-x u or C-_ Undo the previous change, can be repeated.

C-f One character forward.

C-b One character backward.

C-p Go upward one line.

C-n Go down one line.

M-f Move one word forward.

M-b Move one word backward.

C-a Move to the beginning of the line.

C-e Move to the end of the line.

M-< Move to the beginning of the document.

M-> Move to the end of the document.

M-x goto-line Go to line n in the buffer. You will get a prompt asking for the line number.

C-v Page down.

M-v Page up.

C-Space Put the mark here, the current position.

C-k Cut from the marker to the end of line. Repeat for several rows.

C-w Cut the area from the mark to the current position of the marker.

M-w Copy the area from the mark to the current position of the marker.

C-y Paste the latest cut/copy. Can be used repeatedly.

C-s Search forward.

C-r Search backward.

M-x doctor If you find Emacs frustrating.

M-x hanoi If you are bored...

A.3 LaTeX

Here are some of the most common tags that are used when you write a document in LaTeX. Every tag begins with a \ and the environments have a \begin{env} at the beginning and a \end{env} at the end.

A.3.1 Environments

document This is the environment in which the document is set. Everything that is written before this environment begins is called the pre-amble.

abstract Write the abstract within this environment.

equation Will put you in math-mode and allow you to type an equation that you can for reference. The equation will be numbered.

figure Put pictures in this environment.

table You can typeset tables within this environment. Mostly used together with the **tabular**.

tabular Here you order the contents of a table.

thebibliography The environment for the items in your reference list.

A.3.2 Typesetting text

section This is one of the available levels of sectioning. Which of the sectioning commands that are available depends on the specification attached to the **documentclass**.

maketitle Typeset the title, author and date of the document.

tableofcontents Make a table of contents.

emph Typeset the text in an emphasized style, usually italics.

textbf Typeset text in bold font.

caption The text to a table or figure caption.

ref Used to cross-reference a figure, table, or section.

cite Reference a book, article or other bibliographic item from the reference list.

bibitem Marks the items in the reference list.

includegraphics Used when you want to import a picture, ps or eps formatted, into the document. (You will need to include the `graphics`-package in the pre-amble.)

footnote Create a footnote at the bottom of the page containing the text included in the command.

A.3.3 Special symbols

Type	Print
\#	#
\$	$
\%	%
\&	&
\textasciitilde	˜
_	_
\textasciicirsum	ˆ
\textbackslash	\
\{	{
\}	}
@	@
\textquotedblright	”
\textbar	|

A.3.4 Some mathematical symbols

frac Makes a fraction of the two entries.

sum The sign for a sum.

int The integral

alpha Produces the Greek letter α. You can get the other letters from the Greek alphabet by calling them by their name.

pm Plus-minus, $\pm$.

leq Less or equal to, $\leq$.

parallel Parallel to, $||$.

perp Perpendicular to, $\perp$.

hbar The h-bar, Planck’s constant divided by 2π, $\hbar = \frac{h}{2\pi}$.

There are a lot of other tags and environments available. Read the dd-LaTeX home pages and take a look at the commands at
`http://mitra.phys.uit.no/edb/latex/index.html`,
or buy the book “LaTeX – user’s guide and reference manual” by L. Lamport.

A.4 MATLAB

This is a short list of useful command in MATLAB. Consult the MATLAB-help to learn more about how to use them.

quit Quit the MATLAB session. *Remember to always quit MATLAB before you log out!*

help On-line help, display text at command line.

helpwin On-line help, separate window for navigation.

helpdesk Comprehensive hypertext documentation and troubleshooting. **Helpdesk** loads the main MATLAB Help Desk page into the Web browser.

lookfor Search MATLAB for functions that matches some keyword.

format Set the output format.

more Control paged output in command window.

ones Gives you an array of only ones.

zeros Gives you an array of only zeros.

eye Gives you the identity matrix.

rand Gives you an array of uniformly distributed random numbers.

size Gives the dimensions of an array.

length Gives the size of the largest dimension of the array.

eps Floating point relative accuracy. This is the highest accuracy you can get out of MATLAB.

abs Get the absolute values of the elements in an array.

/ Right matrix divide.

./ Right array divide, elementwise.

***** Matrix multiply.

.* Elementwise array multiply.

eig Get the eigenvalues and eigenvectors of a matrix.

diag If you give a vector as the argument you will get a diagonal matrix with that vector as the diagonal, if you give a matrix as the argument you will get the diagonal of that matrix as a vector.

sum Returns the sum of the elements in an array.

fft One-dimensional discrete Fourier transform.

ifft One-dimensional inverse discrete Fourier transform.

fftshift Shift DC component to center of spectrum.

ode45 Solve non-stiff differential equations, medium order method. MATLAB also provides some other methods of solving differential equations.

who List current variables.

whos List current variables with more information.

clear Clear variables and functions from memory. *Warning! There is no undo available!*

plot Linear plot.

xlabel Set the text to be put on the x-axis. To set the text of the other axis there is **ylabel**.

title Set the title of the figure.

mesh 3-D mesh surface plot.

surf 3-D colored surface plot.

subplot Split the figure into several plots.

hist Analyze data to produce a histogram.

bar Draws a bar graph.

print Print a figure to a file, or the printer.

> **print -dps filename** Print the figure to a PostScript-file.
>
> **print -deps -epsi filename** Print the figure in Encapsulated PostScript to filename.
>
> (You can also use the scroll-bar of the figure window to save a figure to a file.)