Contents

List of Figures

List of Tables

Chapter 1

Single time domain signal presentation

1.1 Computational environment setting

1.1.1 Setting up X-Server enabled SSH access under Windows with the initial password "coursework"

The following procedure can be shortened by following the instruction at
`http://www.linux-tip.net/cms/content/view/302/26`.

Prior to starting work on the course materials it is necessary to connect to the Linux machine which stores user programs and data The computer clusters in EEE have only Windows operating systems installed. Hence setting up a remote connection from a Windows to a Linux system is required to access the course materials.

X Window System (X, X11) is a toolkit and a protocol for designing Graphical User Interfaces (GUI) on Unix operating systems (OS). X11 is created as an additional application layer built on top of the OS kernel. X Window System implements a client-server architecture and can be used over a network connection transferring the remote program's output to a local system. In this case a local system will be an X server providing the graphical output for a remote application called an X client. To secure the data communication between the local and remote machines the data exchange has to be tunnelled over the Secure Shell (SSH) protocol.

A software called Exceed X Server will be used as a Windows-based X server to fetch the output from the remote Linux machine. A terminal emulator program named PuTTY could be applied to connect to the Linux machine over the SSH.

Login to Windows from one of the University computer clusters. You should use your ordinary University-wide username and password to login the Windows. Start the Exceed X Server, which can be found in the Start Menu under the following path:

Start → All Programs → Programs Core → Network and Email → Exceed 2006 → Exceed

Refer to Figure 1.1 to find the correct menu option.

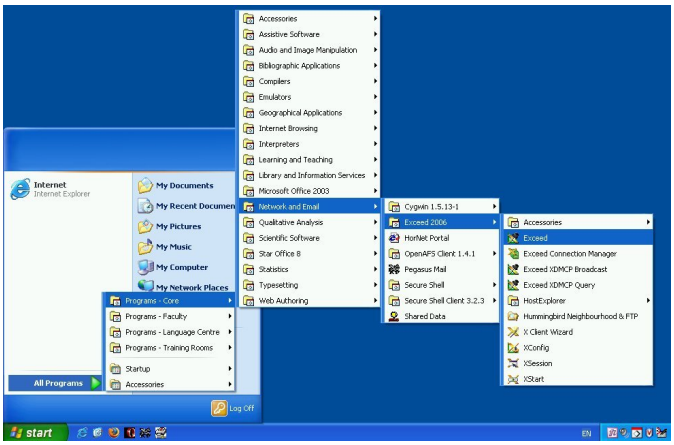


Figure 1.1: Starting Exceed X Server

Now you should launch the terminal emulator PuTTY. It resides in the Start Menu under:

Start → All Programs → Programs Core → Network and Email → Secure Shell → PuTTY

Figure 1.2 shows the exact location of the PuTTY program.

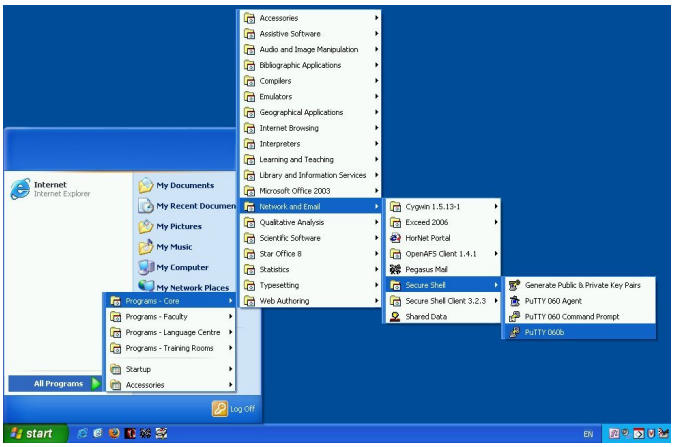


Figure 1.2: Starting PuTTY

Setting the remote machine's IP address and port number is illustrated in Figure 1.3. Type in the Linux system's IP address 130.88.154.34 in the text box "Host Name" in the PuTTY con-

figuration window. Type in the port number 22111 into the field named “Port”. Leave the connection type set to “SSH” by default.

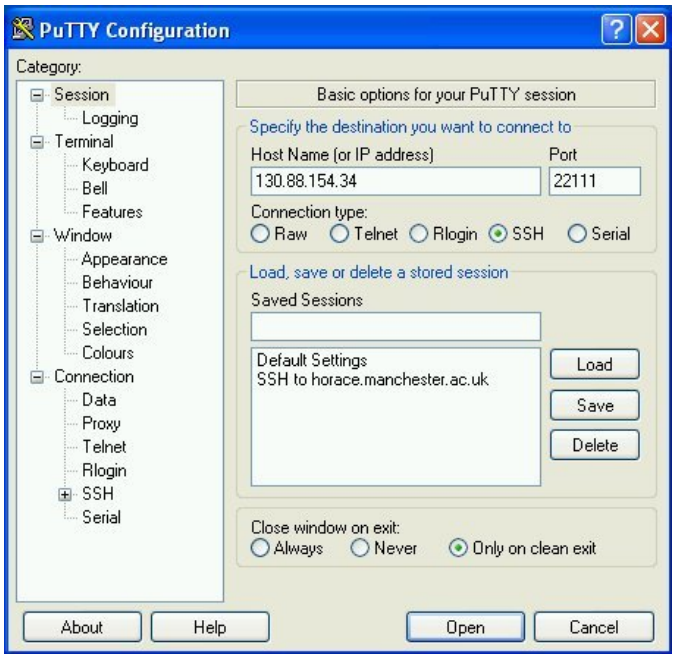


Figure 1.3: Setting IP Address and Port Number in PuTTY

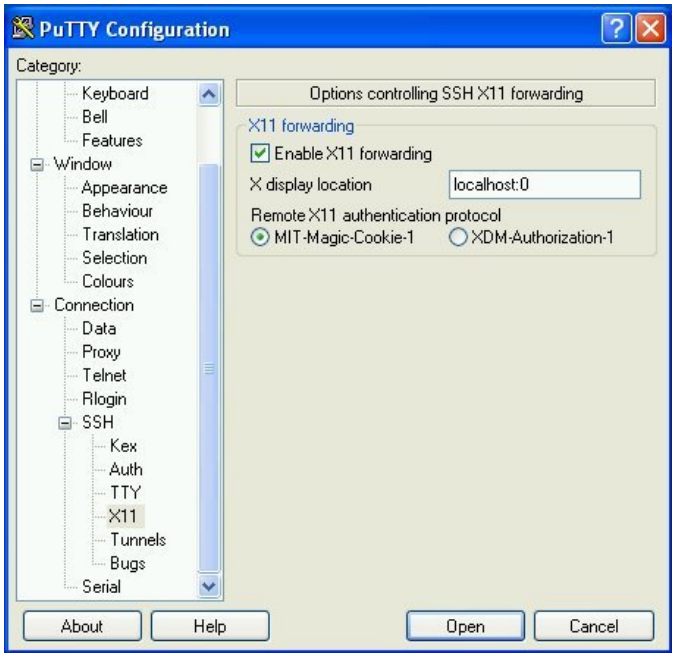


Figure 1.4: Setting X11 Forwarding in PuTTY

Figure 1.4 shows the correct PuTTY settings to enable the X11 forwarding over a secure SSH channel. Expand the “SSH” node in the “Category” list of the PuTTY main configuration window. Then select the “X11” option. Check the box “Enable X11 forwarding” and type in localhost:0 into the text box named “X display location”.

The PuTTY connection settings could be saved for a convenience of future accesses. Figure 1.5 provides guidance on saving the settings. Return to the “Session” node in the “Category” list of the main window. Input the session name, e.g. eee server, into the “Saved Sessions” field and save the settings by pressing the button “Save”. Select the desired session next time you start PuTTY and press the button “Load”. Clicking the button “Open” will establish an SSH connection to a target machine.

Enter your course username and password into the newly appeared terminal window. The first time you connect to a Linux station the host identification message will pop up. Click “Yes” to accept the servers public key and store it the local database. To test that X11 forwarding is setup correctly launch kcalc the KDE calculator application by typing kcalc into a PuTTY terminal. A new calculator window should appear proving the right connection setup. Please put your own password as a new password and re-use the password for the rest of the work during this module.

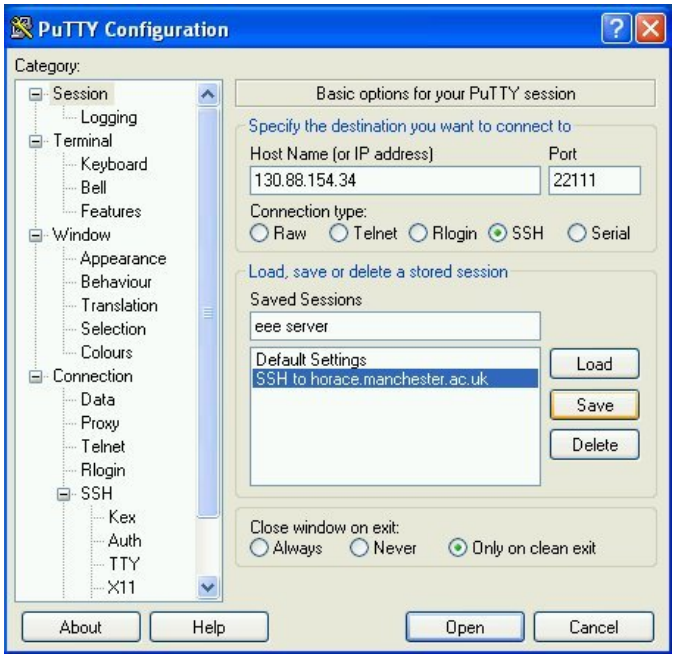


Figure 1.5: Saving Connection Settings in PuTTY

```
[clarissa]gnuplot

        G N U P L O T
        Version 4.0 patchlevel 0
        last modified Thu Apr 15 14:44:22 CEST 2004
        System: Linux 2.6.13-15.10-default

        Copyright (C) 1986 - 1993, 1998, 2004
        Thomas Williams, Colin Kelley and many others

        This is gnuplot version 4.0.  Please refer to the documentation
        for command syntax changes.  The old syntax will be accepted
        throughout the 4.0 series, but all save files use the new syntax.

        Type `help` to access the on-line reference manual.
        The gnuplot FAQ is available from
                http://www.gnuplot.info/faq/

        Send comments and requests for help to
                <gnuplot-info@lists.sourceforge.net>
        Send bugs, suggestions and mods to
                <gnuplot-bugs@lists.sourceforge.net>


Terminal type set to 'x11'
gnuplot>
```

Figure 1.6: Starting point of gnuplot

```
[clarissa]gnuplot

        G N U P L O T
        Version 4.0 patchlevel 0
        last modified Thu Apr 15 14:44:22 CEST 2004
        System: Linux 2.6.13-15.10-default

        Copyright (C) 1986 - 1993, 1998, 2004
        Thomas Williams, Colin Kelley and many others

        This is gnuplot version 4.0.  Please refer to the documentation
        for command syntax changes.  The old syntax will be accepted
        throughout the 4.0 series, but all save files use the new syntax.

        Type `help` to access the on-line reference manual.
        The gnuplot FAQ is available from
                http://www.gnuplot.info/faq/

        Send comments and requests for help to
                <gnuplot-info@lists.sourceforge.net>
        Send bugs, suggestions and mods to
                <gnuplot-bugs@lists.sourceforge.net>


Terminal type set to 'x11'
gnuplot> plot [-1:6][0:16] 'datafile1' w lp ps 10 lw 3
```

Figure 1.7: Plotting a datafile

1.2 Graph production

Gnuplot is one of the simple and versatile scientific tools to produce a graph from the data. Reference: `http://www.sns.ias.edu/computing/pages/linux/gnuplot/gnuplot.html#8684` In general on programming, see `http://en.wikipedia.org/wiki/Programming_style` and if you are stuck, you could place your question at `http://www.edaboard.com/`

1.2.1 Datafile

An ascii data file `datafile1` which contains

```
1
2
3
4
5
```

can be produced by invoking `emacs` or some other software. In the similar way, another 2 files called `datafile2` which contains

```
6
7
8
9
10
```

and `datafile3` which contains

```
11
12
13
14
15
```

can be produced.

When you run a command `gnuplot`, you get Fig. 1.6. When you type plot command as is shown in Fig. 1.7, you will get a window of Fig. 1.8 Here, `[-1:6][0:16]` means x and y plot ranges are from -1 to 6 and from 0 to 16, respectively. `w lp ps 10 lw 3` means plotting data **w**ith **l**ine**p**oints and the **p**oint**s**ize is 10 and **l**ine**w**idth is 3. More than one lines can be plotted in one window as is demonstrated in Fig. 1.9 and the resultant figure is in Fig. 1.10. So far, the data are all one column and the value of x-axis is pre-set from 0. However, when there are two columns, the information on first column is used as x-axis. This can be tested as follows: `paste datafile2 datafile3 > datafile4` The `datafile4` has two columns which can be checked by `less datafile4` . When Fig. 1.11 is typed as gnuplot command line, Fig. 1.12 is obtained. . Produce a 15×1 vector by

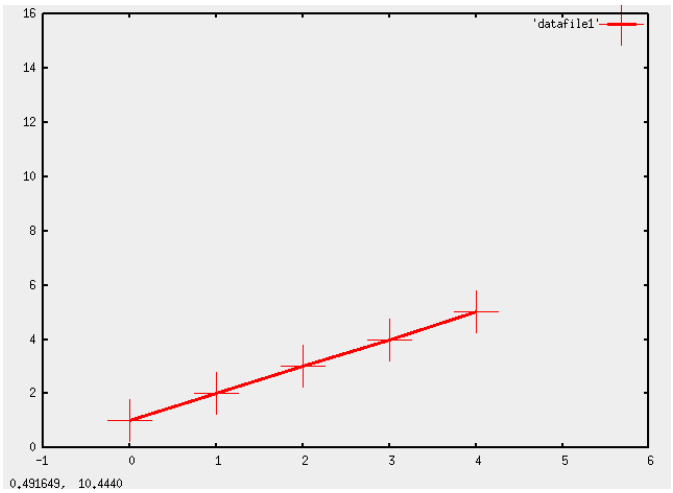


Figure 1.8: Example figure for datafile1

```
[clarissa]gnuplot

        G N U P L O T
        Version 4.0 patchlevel 0
        last modified Thu Apr 15 14:44:22 CEST 2004
        System: Linux 2.6.13-15.10-default

        Copyright (C) 1986 - 1993, 1998, 2004
        Thomas Williams, Colin Kelley and many others

        This is gnuplot version 4.0.  Please refer to the documentation
        for command syntax changes.  The old syntax will be accepted
        throughout the 4.0 series, but all save files use the new syntax.

        Type `help` to access the on-line reference manual.
        The gnuplot FAQ is available from
                http://www.gnuplot.info/faq/

        Send comments and requests for help to
                <gnuplot-info@lists.sourceforge.net>
        Send bugs, suggestions and mods to
                <gnuplot-bugs@lists.sourceforge.net>


Terminal type set to 'x11'
gnuplot> plot [-1:6][0:16] 'datafile1' w lp ps 10 lw 3,'datafile2' w lp ps 6
lw 2
gnuplot>
```

Figure 1.9: Plotting two datafiles, showing the command line

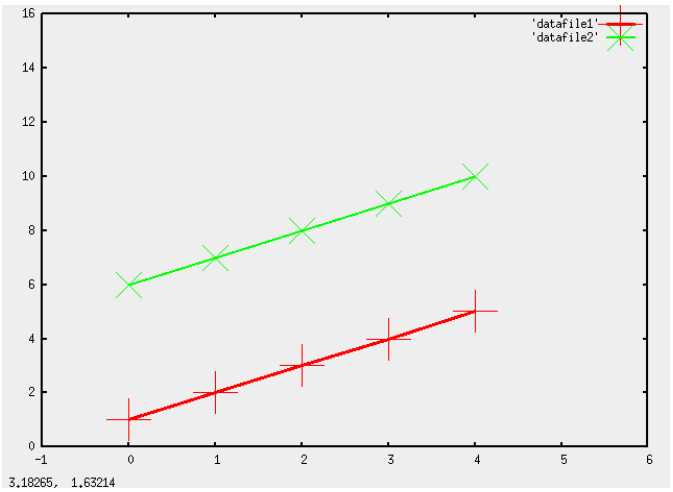


Figure 1.10: Gnuplot presentation as a result of Fig. 1.9

```
[clarissa]gnuplot

        G N U P L O T
        Version 4.0 patchlevel 0
        last modified Thu Apr 15 14:44:22 CEST 2004
        System: Linux 2.6.13-15.10-default

        Copyright (C) 1986 - 1993, 1998, 2004
        Thomas Williams, Colin Kelley and many others

        This is gnuplot version 4.0.  Please refer to the documentation
        for command syntax changes.  The old syntax will be accepted
        throughout the 4.0 series, but all save files use the new syntax.

        Type `help` to access the on-line reference manual.
        The gnuplot FAQ is available from
                http://www.gnuplot.info/faq/

        Send comments and requests for help to
                <gnuplot-info@lists.sourceforge.net>
        Send bugs, suggestions and mods to
                <gnuplot-bugs@lists.sourceforge.net>

Terminal type set to 'x11'
gnuplot> plot [-1:12][0:16] 'datafile4' w lp ps 10 lw 3
```

Figure 1.11: Plotting two column data

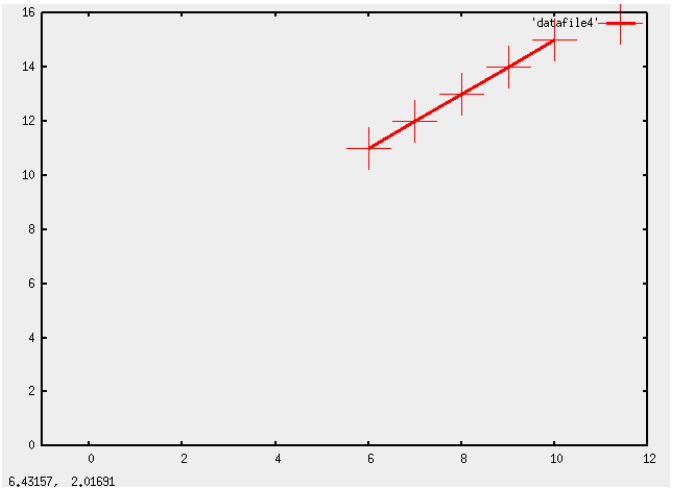


Figure 1.12: Gnuplot presentation as a result of Fig. 1.11

```
gnuplot> set samples 10
gnuplot> plot 'datafile5'  smooth bezier w lp ps 10,'datafile5'  w l
```

Figure 1.13: Example of smoothing in gnuplot

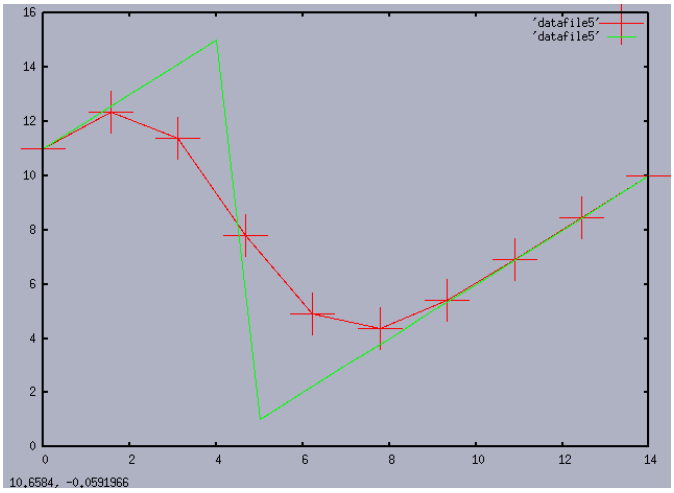


Figure 1.14: Gnuplot presentation as a result of Fig. 1.13

`cat datafile3 datafile1 datafile2 > datafile5`
and run gnuplot as in Fig. 1.13 then Fig. 1.14 will be obtained. This smoothing function is useful and this smoothed data can be saved in an ascii file by typing like Fig. 1.15 A file `tableoutput` as is shown in Fig. 1.16. The first 10 lines are on the bezier smoothing curve and the second 15 lines are the original data whose second column should be the same as `datafile5`.

Gnuplot has online help. The command help as is shown in Fig. 1.17 gives explanation as well as example usage. A variety of gnuplot commands will be learnt from the online help.

```
gnuplot> set samples 10
gnuplot> set terminal table
Terminal type set to 'table'
gnuplot> set output 'tableoutput'
gnuplot> plot 'datafile5'  smooth bezier w lp ps 10,'datafile5'  w l
```

Figure 1.15: Saving smoothed data

```
[clarissa]more  tableoutput
#Curve 0, 10 points
#x y type
 0  11  i
 1.55556  12.3411  i
 3.11111  11.3799  i
 4.66667  7.79917  i
 6.22222  4.89082  i
 7.77778  4.36133  i
 9.33333  5.39393  i
 10.8889  6.89068  i
 12.4444  8.44445  i
 14  10  i

#Curve 1, 15 points
#x y type
 0  11  i
 1  12  i
 2  13  i
 3  14  i
 4  15  i
 5  1  i
 6  2  i
 7  3  i
 8  4  i
 9  5  i
 10  6  i
 11  7  i
 12  8  i
 13  9  i
 14  10  i

[clarissa]
```

Figure 1.16: Gnuplot saving as a result of Fig. 1.15

```
gnuplot> help splot
```

Figure 1.17: Gnuplot help usage

1.3 Assignment on gnuplot for EEEN40063: Students on EEEN60141 should perform assignments 3,4,5,6, and 7

If some problems with Unix commands are encountered, see `http://www.tuxfiles.org/linuxhelp/cli.html` for more information on Unix commands such as

diff	cat	cp	file	head
less	ln	ls	mkdir	mv
pwd	rm	tail	touch	wc
grep	whereis	which	echo	emacs
gawk	paste	sort	bc	clear
man	passwd	bzip2	gunzip	gzip
tar	date	df	du	hostname
kill	ps	top	who	whoami

Pay extra attention to the handout; some characters such as `,` and `.` and `'` and `` ` `` are difficult to distinguish in printing and using wrong characters lead to considerable delay in code debugging.

1. Run a command `script` at the beginning of your work under Linux. This as this command will make a log of your usage of unix environment and this log will be used for the part of the assessment.

2. Create a directory called `assignment1` by `mkdir assignment1` under your home directory, move to the directory by `cd assignment1`.

3. Produce a three dimensional `datafile` of your own by `emacs datafile` (i.e., each row must have x-value, y-value). Name the file `datafile`. Please set the dimension of $x \times y$ as more than 2×2. The range of x and the range of y should be different and the dimension of x and the dimension of y should be different such as 2×4.

4. Produce a contour figure with gnuplot. The following gnuplot functions could be used.

   - nosurface
   - view
   - contour
   - cntrparam
   - dgrid3d
   - splot

   Launching a command `help set` will sow more details of the functions.

5. Save the contour figure by running the following commands

   ```
   /usr/X11R6/bin/xwd -screen  > a.xwd
   (choose the frame you would like to save)
   convert  a.xwd  a.pdf
   ```

   Name the contour figure `contourfigure.pdf`. These pdf files will be assessed.

6. Send these pdf files as well as `datafile` to your own University email account and assemble your assignment report with these files.
   The example command to send a file is
   ```
   mail -r f.costen@cs.man.ac.uk
   -a contourfigure.pdf
   f.costen@cs.man.ac.uk
   ```
   You should change the email address to your own email address and to send the email, you finish the command with `cntl + d`. If emailing does not work, do the followings on the figure window:

   (a) click on left handside top

   (b) select "Edit"

   (c) select "Copy All"

   (d) select "To file"

   to save the figure on your Window side.

7. Type `exit` at the end of your work. This will produce a file called `typescript` in the current directory. Change the name of the typescript to, for example, typescript-lab1 which should be in `/YourHomeDirectory/assignment1/`, not any other subdirectories such as `/YourHomeDirectory/assignment1/subdirectory`.
   When you work on the project more than once, please run `script` at the beginning of your work and finish with `exit` at the end of the session and perform
   ```
   cat typescript >>
   /YourHomeDirectory/assignment1/typescript-lab1
   ```
   to accumulate your log under linux.

The assignment should be compiled with the essence of your `typescript` during your work with `gnuplot`. This `typescript` will be the part of assessment.

Assessment criteria are as follows:

- appropriate number of line width
- appropriate contour increment
- appropriate view point
- appropriate x tics and y tics
- appropriate xlabel and ylabel

Finally, if you would like to use Latex for your documentation, you may obtain an introductory information about Latex in `http://www.ctan.org/tex-archive/info/lshort/english/lshort.pdf`, and `http://it.metr.ou.edu/latex/`

Figure 1.18: Example answer for datafile

– The range of x is different from the range of y.

[1 mark]

– The dimension of x is different from the dimension of y

[1 mark]

The assignment report report and typescript for gnuplot part should have similar figures to Fig. 1.18, Fig. 1.19, and Fig. 1.20.

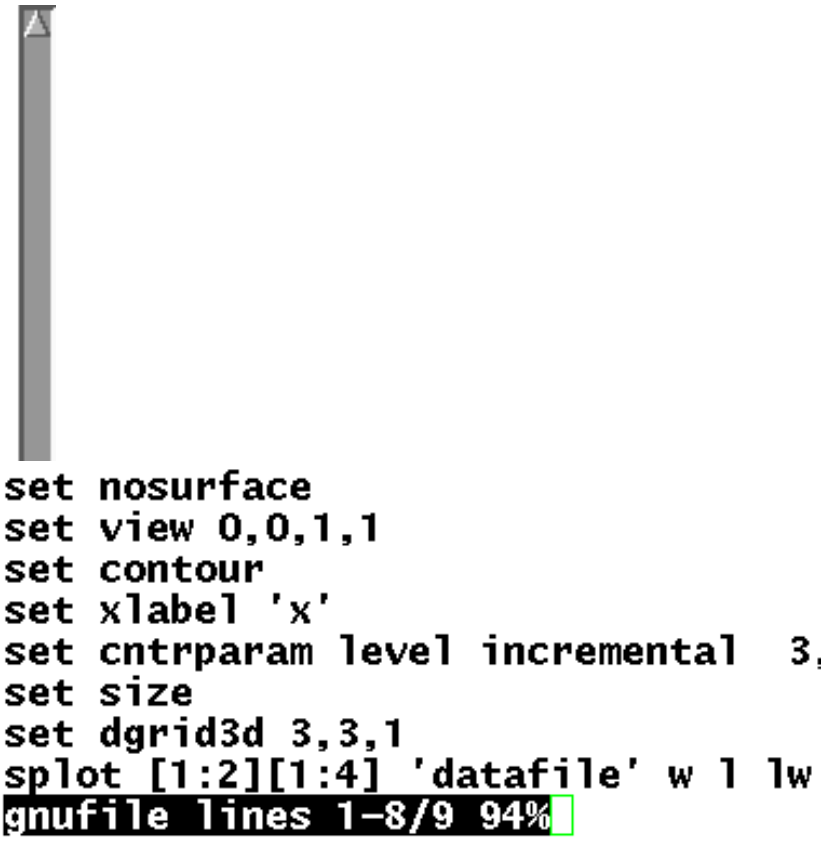


Figure 1.19: Example answer commandwindow.pdf

– Proper usage of view

[1 mark]

– Proper usage of cntrparam

[2 marks]

– Proper usage of dgrid3d

[2 marks]

– Proper usage of splot

[1 mark]

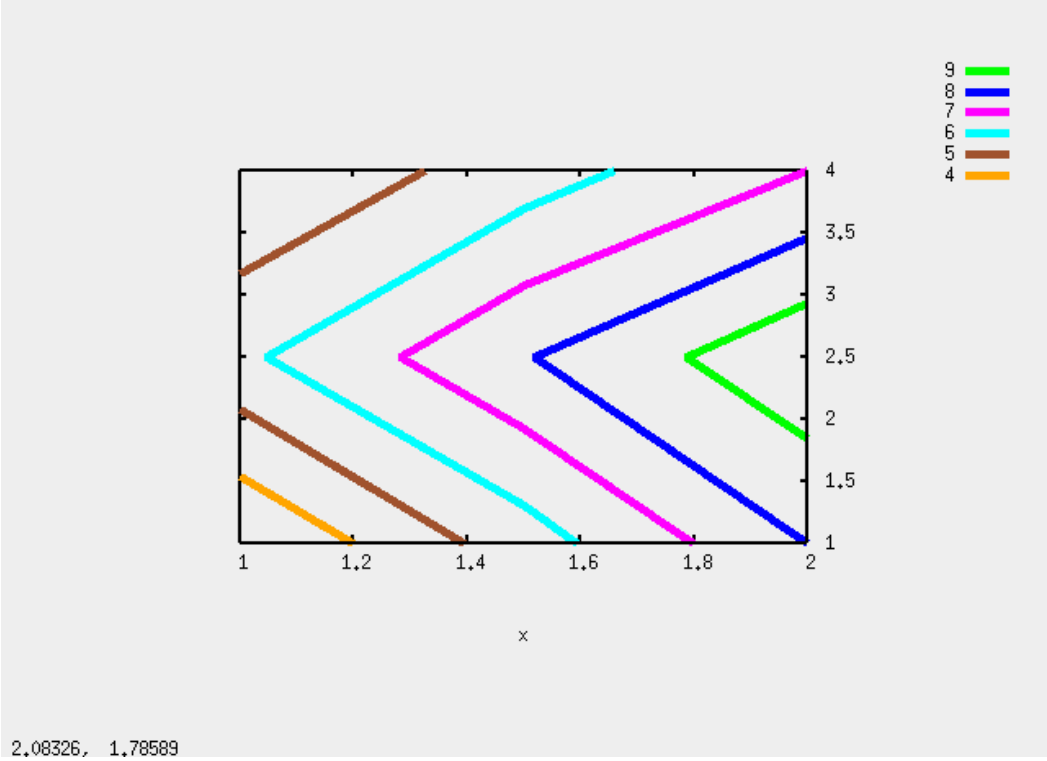


Figure 1.20: Example answer contourfigure.pdf
The datafile which students produce should be tested using typerscript for gnuplot part which students produce to see if the figure the assessor produces matches to contourfigure.pdf which students produce.

[2 marks]

Submission of the assignment for EEEN40063

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Student signature	

.....

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Recipient signature	

Chapter 2

Multiple time domain signal presentation

```
gnuplot> set terminal tgif portrait  solid  "Times-Roman" 10
Terminal type set to 'tgif'
Options are 'portrait [1,1] solid "Times-Roman" 10'
gnuplot> set output 'test.obj'
gnuplot> plot 'datafile4' u 2:1 w l
```

Figure 2.1: Gnuplot commands to set the terminal to tgif

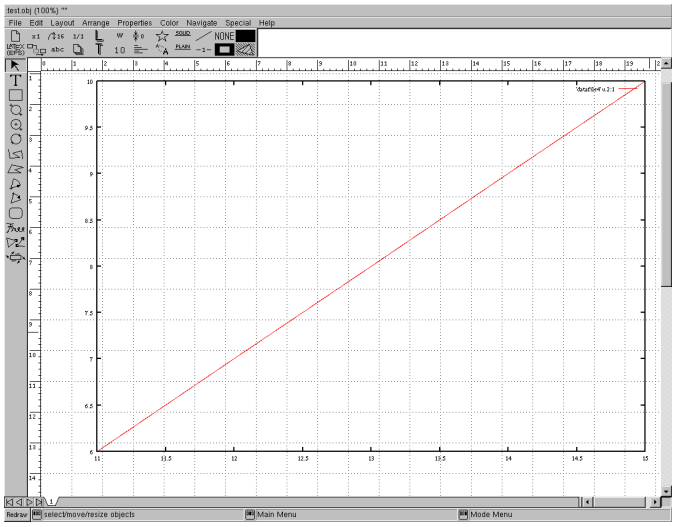


Figure 2.2: Tgif main window with gnuplot data

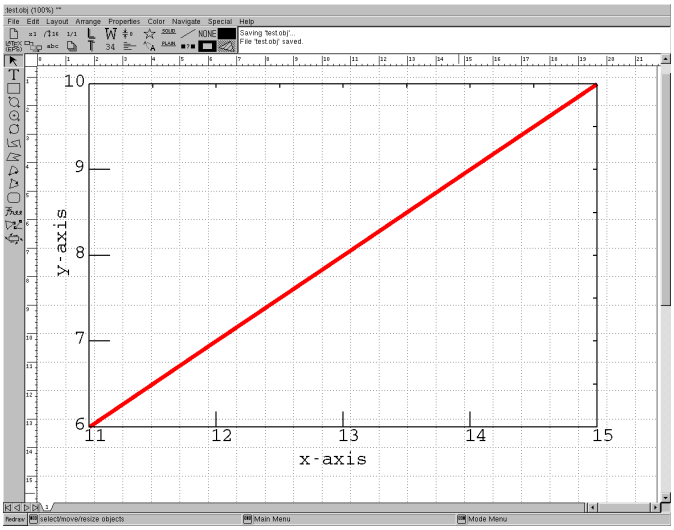


Figure 2.3: Decoration of Fig. 2.2 in tgif

2.1 Decoration of scientific figures

There is a standard drawing tool called `tgif` in Unix and Linux environment and `gnuplot` has a terminal to `tgif`. Figures produced by `gnuplot` can be saved in tgif format (`.obj`) for the postprocessing for better presentation. For example, the commands in Fig. 2.1 will produce `test.obj`. This `test.obj` can be open by `tgif test.obj`. `tgif` opens a new window shown in Fig. 2.2. The graph can be decorated like Fig. 2.3 where x, y axis are labelled and ticked properly and lines are made thick. This file can be turned into pdf file as follows:

```
make -f  tgif-eps.mak
epstopdf test.eps -o=test.pdf
```

when `tgif-eps.mak` is in the working directory. The main functions usually used for scientific presentation are pointed out in Fig. 2.4.

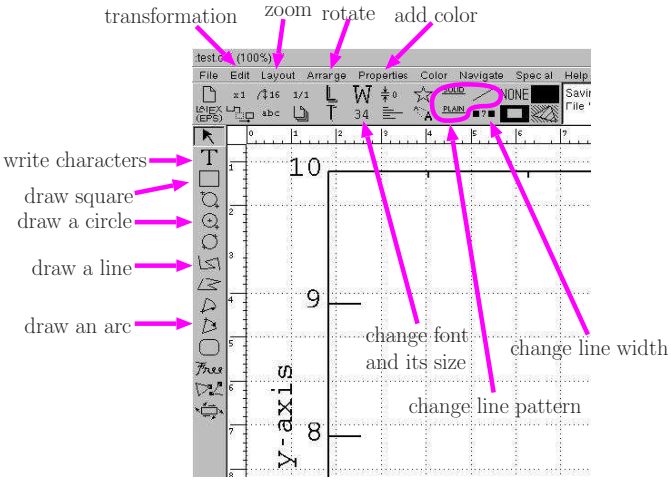


Figure 2.4: Major tgif functions

Table 2.1: The allocation of the columns each user should use

UserNo.	column number for plotting
1	1 , 2 , 6
2	1 , 2 , 7
3	1 , 3 , 4
4	1 , 4 , 6
5	2 , 3 , 7
6	2 , 3 , 8
7	2 , 4 , 5
8	2 , 4 , 7
9	2 , 5 , 6
10	2 , 5 , 7
11	2 , 5 , 8
12	3 , 4 , 8
13	3 , 5 , 6
14	3 , 5 , 8

UserNo.	column number for plotting
15	3 , 6 , 7
16	3 , 6 , 8
17	4 , 6 , 7
18	4 , 7 , 8
19	5 , 7 , 8
20	1 , 2 , 8
21	1 , 3 , 6
22	1 , 3 , 8
23	1 , 4 , 5
24	1 , 4 , 7
25	1 , 4 , 8
26	1 , 5 , 8
27	1 , 6 , 7
28	2 , 3 , 4

UserNo.	column number for plotting
29	2 , 3 , 5
30	2 , 3 , 6
31	2 , 6 , 7
32	2 , 7 , 8
33	3 , 4 , 5
34	3 , 4 , 6
35	3 , 4 , 7
36	3 , 7 , 8
37	4 , 5 , 6
38	4 , 5 , 7
39	4 , 5 , 8
40	5 , 6 , 7
41	5 , 6 , 8
42	6 , 7 , 8

UserNo.	column number for plotting
43	1 , 5 , 6
44	1 , 6 , 8
45	1 , 7 , 8
46	2 , 4 , 6
47	2 , 4 , 8
48	2 , 6 , 8
49	3 , 5 , 7
50	4 , 6 , 8
51	1 , 2 , 3
52	1 , 2 , 5
53	1 , 3 , 5
54	1 , 5 , 7
55	1 , 2 , 4
56	1 , 3 , 7

2.2 Assignment on tgif

Run a command `script` at the beginning of your work under Linux.

A file `/coursedisk/course/skeleton/tgifdata.gz` has 10 column data.

```
cp /coursedisk/course/skeleton/tgifdata.gz .
gunzip tgifdata.gz
```

will unzip the datafile.

Create a directory called `assignment2` under your home directory and move to the directory.

Using the datafile,

1. Plot 3 lines in one figure from the first 8 columns Table 2.1 shows which columns should be used by each user. For example, if your user number is 10, you should use the second column, 5th column and 7th column in the file to produce a figure with three lines.
   Produce a figure with an appropriate legend, axis titles and ticks and convert the tgif file as `3linesfig.pdf` .
2. Use the data in 9th column as x-axis values and use the data in 10th column as the y-axis data and plot two lines , one is original line and the other is the original data scaled by a in one figure. a should be calculated using the formula of a = username number $\times$ 1.2.
3. Apply the tgif treatment. Here, x-axis is the time step and y-axis is the voltage.
4. Convert the tgif file as `2linesfig.pdf`

Type `exit` at the end of your work. This will produce a file called `typescript` at the current directory. Change the name of `typescript` to, for example, typescript-chap2 which should be in `/YourHomeDirectory/assignment2/`, not any other subdirectories such as `/YourHomeDirectory/assignment2/subdirectory`.

Compile your assignment with the commands you used in gnuplot extracted from typescript, `3linesfig.pdf`, typescript, `2linesfig.pdf`.

```
/tmp/tmp6
gnuplot> set terminal tgif solid
Terminal type set to 'tgif'
Options are 'portrait [1,1] solid "Helvetica" 18'
gnuplot> set output '3linesfig.obj'
gnuplot> plot 'tgifdata' u :1 w l ,'tgifdata' u :2 w l ,'tgifdata' u :3 w l
gnuplot>
```

Figure 2.5: Example answer of 3linescommand.pdf

– Appropriate usage of terminal setting

[1 mark]

– Appropriate usage of plot for multiple lines

[1 mark]

The assignment report should have similar answers to Fig. 2.5, Fig. 2.6, Fig. 2.7, and Fig. 2.8.

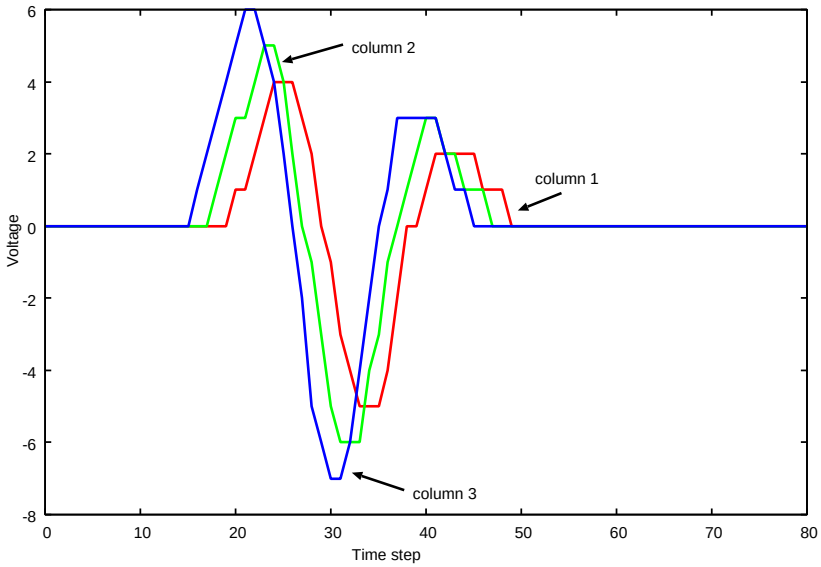


Figure 2.6: Example answer of 3linesfig.pdf

– Appropriate usage of data. Each student has a different dataset to deal with. Make sure the right data is used

[1 mark]

– Appropriate legend

[1 mark]

– Appropriate line thickness

[1 mark]

– Appropriate x, y axis titles

[1 mark]

```
gnuplot> set terminal tgif solid
Terminal type set to 'tgif'
Options are 'portrait [1,1] solid "Helvetica" 18'
gnuplot> set output '2linesfig.obj'
gnuplot> plot 'tgifdata' u 9:10 w l,  'tgifdata' u 9:($10*1.2) w l
gnuplot>
```

Figure 2.7: Example answer of 2linescommand.pdf

– Appropriate usage of terminal setting

[1 mark]

– Appropriate usage of “plot” for data multiplication

[1 mark]

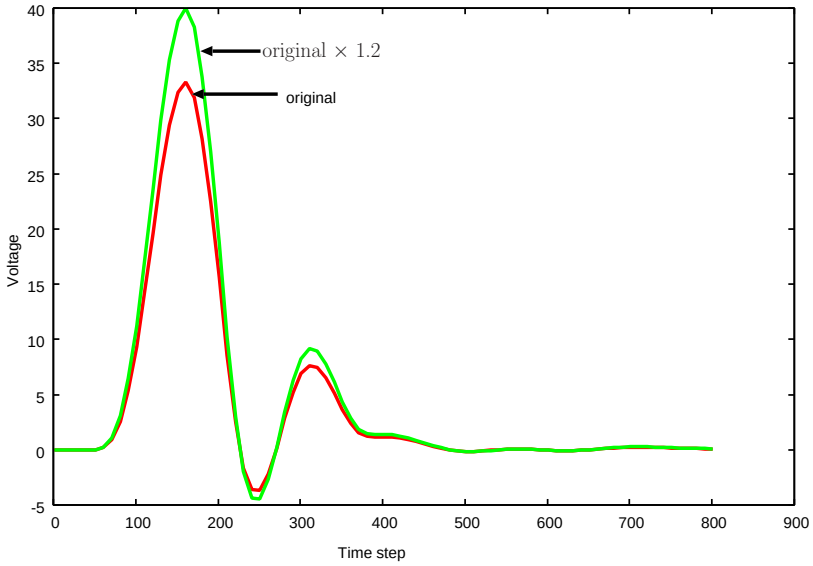


Figure 2.8: Example answer of 2linesfig.pdf

– Appropriate usage of legend

[1 mark]

– Appropriate setting of line thickness, x-, and y- axis titles

[1 mark]

Submission of the assignment for EEEN40063

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Student signature	

.....

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Recipient signature	

Chapter 3

Static spatial domain signal presentation for publication

3.1 Two dimensional data presentation in gray scale:pgm format

`xv` is a software which interprets a variety of two dimensional data. When a datafile `test.pgm` is depicted in Fig. 3.1,

A command `xv -geometry 300x300 test.pgm`

produces a data interpretation shown in Fig. 3.2. The first line of Fig. 3.1 means the data is pgm format and the third line means x-axis has 2 pixels and y-axis has 4 pixels. The fourth line means the effective maximum value in the data is 8. The rest of the data is the actual data at each pixel. The difference in data files shown in Fig. 3.1 and Fig. 3.3 is the third line. This means the way to map the one dimensional data into two dimensional data is different between Fig. 3.1 and Fig. 3.3. Fig. 3.3 is displayed using 4 pixels in X-axis and 2 in Y-axis. When the value of the fourth line is set to a value higher than the maximum of the whole data like Fig. 3.5, the appearance becomes similar to Fig. 3.6 and the entire data is presented as a low value data. On the other hand, when the value of the fourth line is set to one lower than the maximum of the whole data like Fig. 3.7, the data values above 5 are regarded as zero as is shown in Fig. 3.8. The minimum value is presented as black and the maxim value is presented as white in pgm.

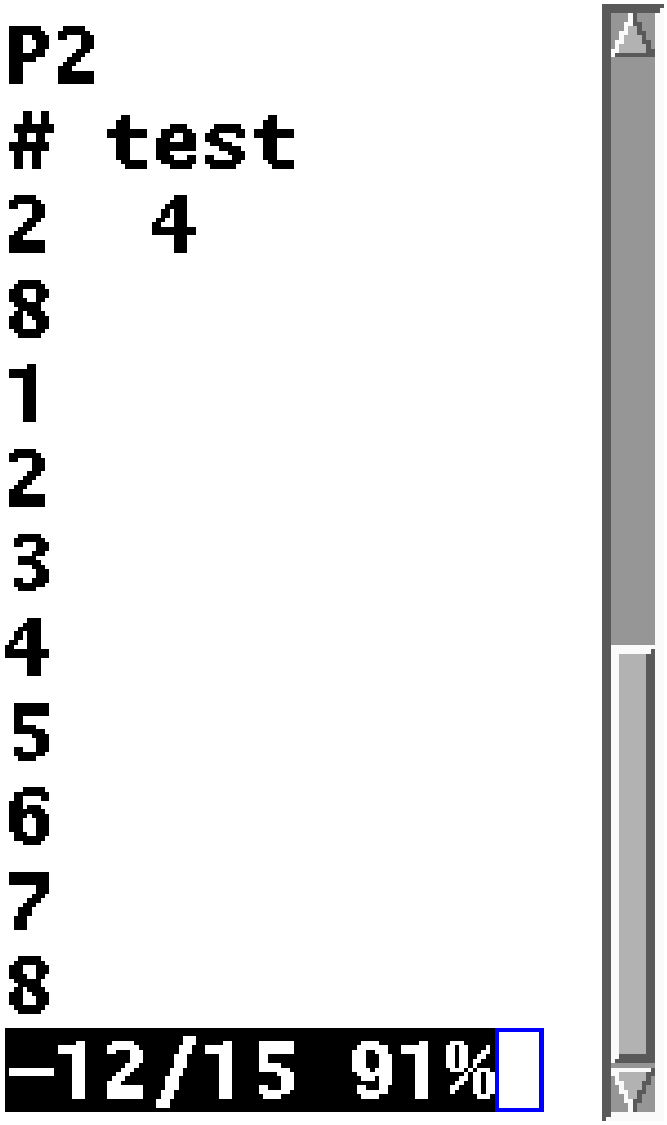


Figure 3.1: Ascii contents of a pgm data file
The data in pgm format can be separated by . or space or return. This means that the data can not be real. The data has to be integer.

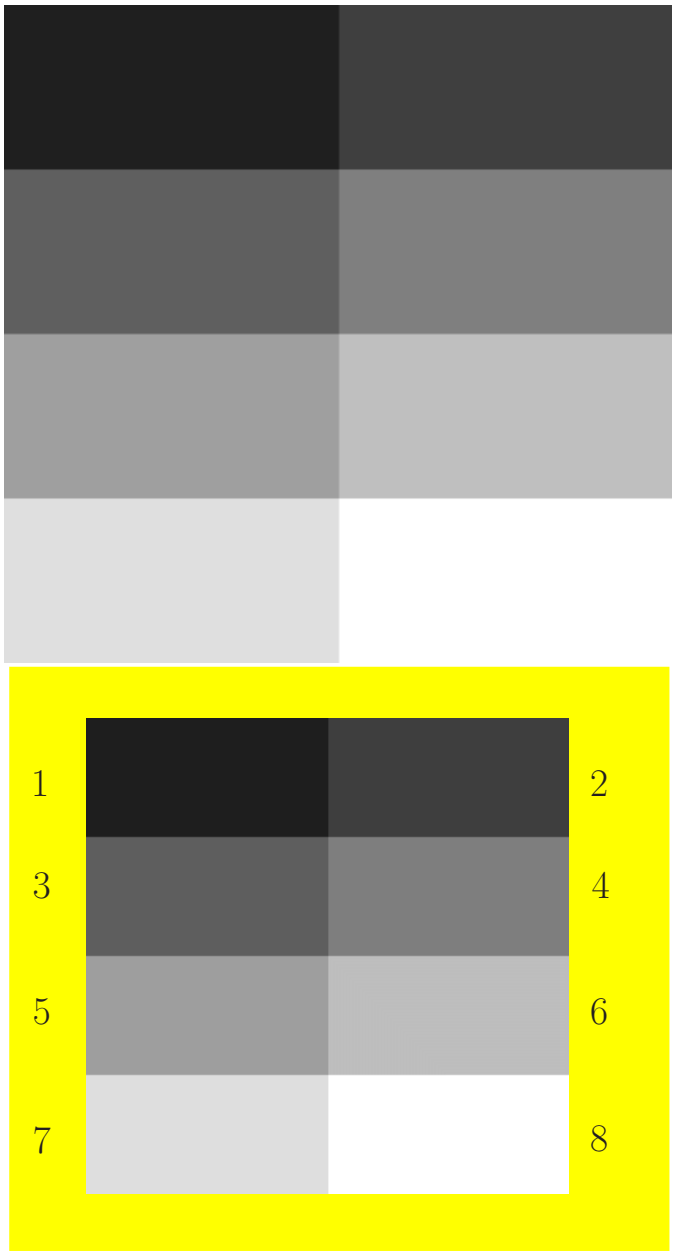


Figure 3.2: Result of Fig. 3.1 in xv

```
P2
# test
4 2
8
1
2
3
4
5
6
7
8
-12/15 91%
```

Figure 3.3: Ascii data format for pgm; similar to Fig. 3.1 but third line is different.

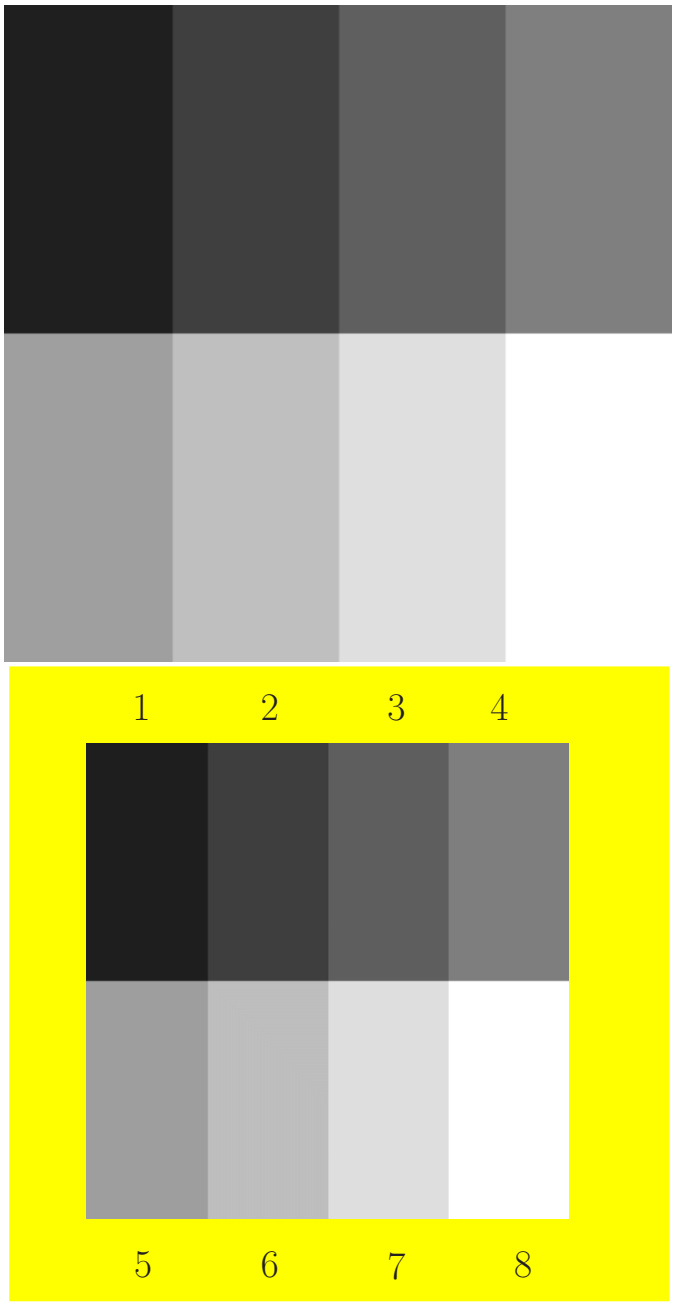


Figure 3.4: Result of Fig. 3.3 in xv

```
P2
# test
4 2
20
1
2
3
4
5
6
7
8
-12/15 92%
```

Figure 3.5: The pgm data file with high setting of maximum value at the fourth line

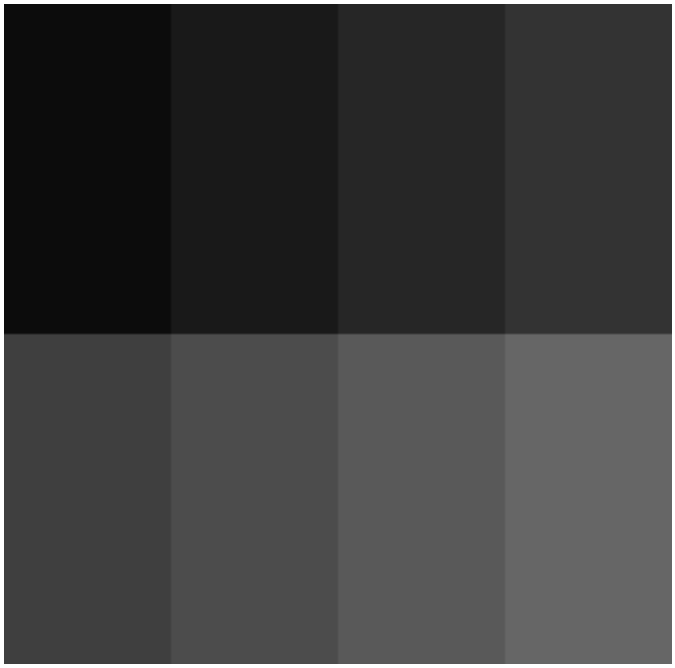


Figure 3.6: Result of Fig. 3.5 in xv

```
P2
# test
4 2
5
1
2
3
4
5
6
7
8
-12/15 91%
```

Figure 3.7: The pgm data file with low setting of maximum value at the fourth line

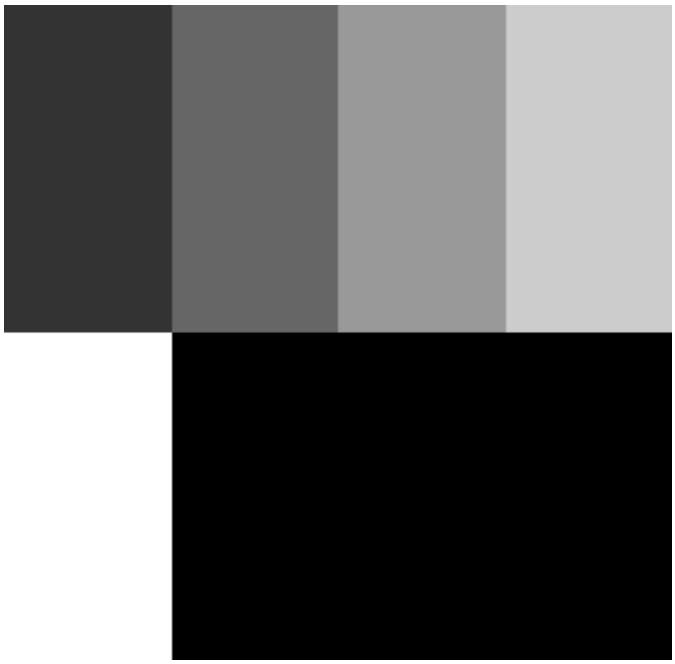


Figure 3.8: Result of Fig. 3.7 in xv

3.2 Assignment on pgm

There are three files called `pgmdata1`, `pgmdata2` and `pgmdata3`. Each datafile should be used to produce one pgm file. x-axis has 5 pixels and y-axis has 3 pixels.

1. Create a directory called `assignment3` under your home directory and move to the directory. Unzip the datafiles `pgmdata?.gz` into the current directory
   ```
   mkdir assignment3
   cd  assignment3
   cp /coursedisk/course/skeleton/pgmdata1.gz
   .  ; gunzip pgmdata1.gz
   ```
   Do the same for `pgmdata2.gz` and `pgmdata3.gz`

2. Modify the datafiles of `pgmdata?` as follows

   - `pgmdata1` should be multiplied by a. Here a is the username number. If your username is eee98, this is achieved by
     ```
     cat pgmdata1 |  awk '{print $1*98}'
      >  pgmdata1-new
     ```
     for example.
   - `pgmdata2` should be multiplied by $(a + 1)$ If your username is eee98, this is achieved by
     ```
     cat pgmdata2 |
     awk '{print $1*99}'
     >  pgmdata2-new
     ```
     for example.
   - `pgmdata3` should be multiplied by $(a + 2)$

3. Produce ascii pgm files from the modified pgmdata files and name them `test1.pgm`, `test2.pgm` and `test3.pgm`, respectively. In setting the maximum value which is fourth line in pgm file, try more than one values to find out the appropriate value for the fourth line(maximum value) in order to obtain the clear figure.

4. Save these three figures as the clip of the window and name them as `test1.pdf`, `test2.pdf` and `test3.pdf`

5. Discuss your observation of `test?.pdf` in a file called `pgmobservation`.

Assemble your assignment report from the ascii files of `test1.pgm`, `test2.pgm`, and `test3.pgm` and the outcome of `xv` *i.e.,* `test1.pdf`,`test2.pdf`,`test3.pdf`. Please use `display` command to view the pgm files as well as `xv` because some cases are trapped by a bug of `xv` and they show nice and clear image with `xv` which should not be the case.

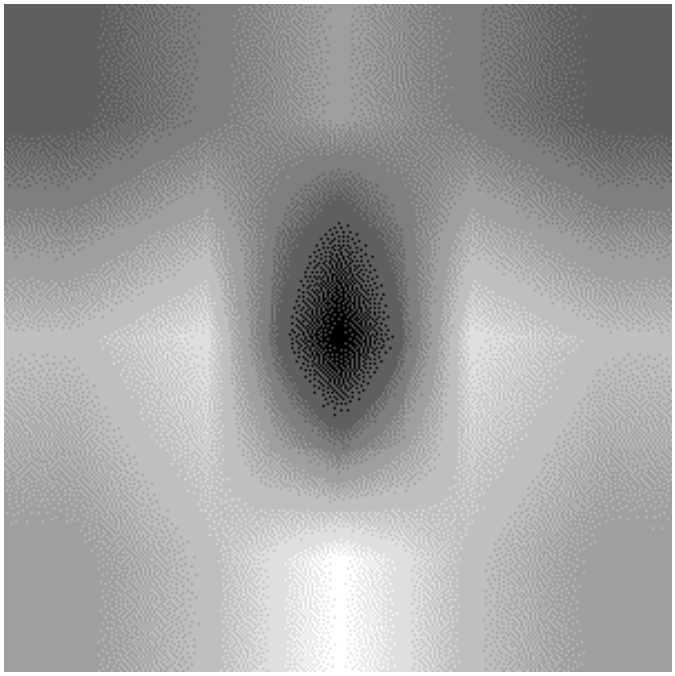


Figure 3.9: Example answer for test1.pgm

[1 mark]

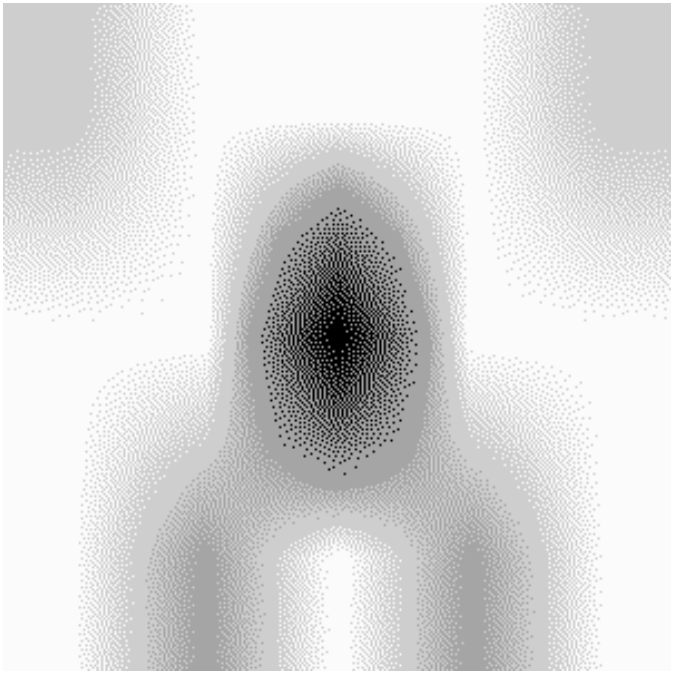


Figure 3.10: Example answer for test2.pgm

[1 mark]

In assignment report, the properly modified data should be presented

[1 mark]

The discussion should state that

– pgmdata1 can be observed properly if the maximum value is set to 8$\times a$

[2 marks]

– pgmdata2 can be slightly observed if the maximum value is set to 308$\times(a+1)$

[2 marks]

– pgmdata3 can not be viewed clearly because all the values are very similar relative to the maximum value even when the maximum value is set to 406$\times(a+2)$

[2 marks]

Figure 3.11: Example answer for test3.pgm

[1 mark]

Submission of the assignment for EEEN40063

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Student signature	

.....

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Recipient signature	

Chapter 4

Stational spatial domain signal presentation in color

```
P3
 # test
              4               2
 255
 0 0 251
 0 119 255
 0 247 255
 115 255 139
 243 255 11
 255 139 0
 255 11 0
 127 0 0
st.ppm lines 1-12/12 (END)
```

Figure 4.1: Ascii contents of a ppm data file converted from Fig. 3.3

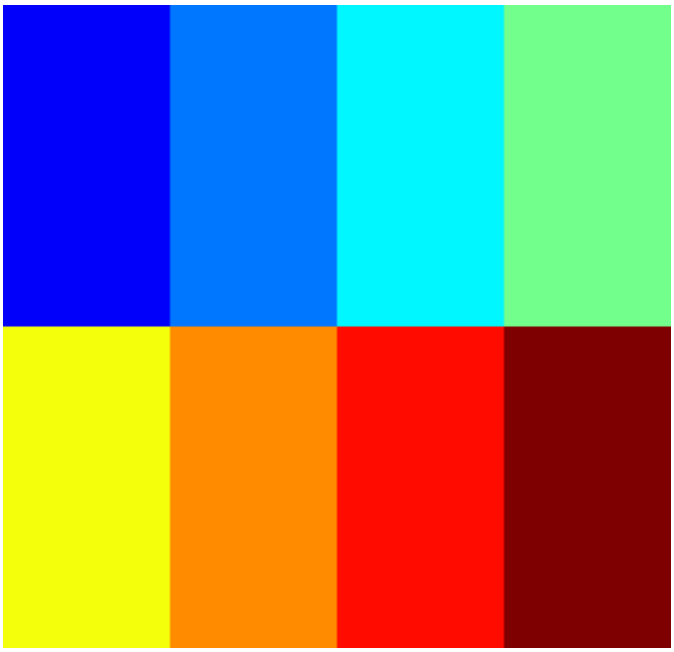


Figure 4.2: Graphical interpretation of a ppm datafile shown in Fig. 4.1 using xv

4.1 Two dimensional data presentation in color

The gray scale `pgm` format can be easily converted to the color `ppm` format by a program called `/coursedisk/course/skeleton/pgm2ppm` Please be aware that the maximum value in pgm file has to be below 1000000000 and negative values are not handled.

If `test1.pgm` is converted, `cp test1.pgm test.pgm; /coursedisk/course/skeleton/pgm2ppm; cp test.ppm test1.ppm` will produce the ppm file `test1.ppm`. For example, when the information in Fig. 3.3 is put into `test.pgm` and run `./pgm2ppm`, Fig. 4.1 will be obtained. The first line of P3 means the file is ppm format, not pgm format and the data from the 5th line to the 12th line are $R_{ed}G_{reen}B_{lue}$ values for each one dimensional (*i.e.,* gray scale) data. The fourth line is the maximum value of all the data from the 5th line to the 12th line. `pgm2ppm` set the maximum value to 255. Unfortunately, there is a single space before a word `P3` in the first line. Both pgm and ppm formats are sensitive to the space only in the first line and this single space in front of `P3` or `P2` prevents us from viewing the file via `xv`. To remove this space, `cat test.ppm | awk '(NR == 1){print $1 }(NR != 1){print $0}' > test.ppm` This command means if the line number is 1, print the first field of $1 and if the line number is not 1, print all the values in the lines. More information on awk can be obtained by executing `man awk` .

The following command `xv -geometry 300x300 test.ppm` will produce Fig. 4.2. The highest color is brown and the lowest color is mapped as black. It is not wise to play with the value of the fourth line. If the value of the fourth line is reduced to , say, 200, the values which are greater than 200 from the 5th to 12th line are regarded as zero.

The complete color range is obtained by running `/coursedisk/course/skeleton/pgm2ppmtmp` which produces the full range of the color `test.ppm` without any input as is shown in Fig. 4.3. Fig. 4.2 can be saved as a gif file. Clicking the right button of your mouse on the

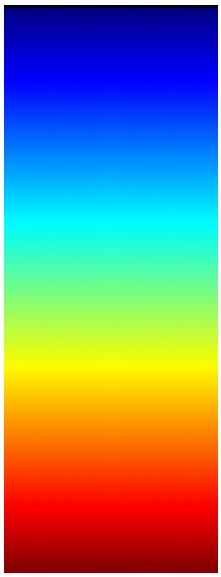


Figure 4.3: Color bar

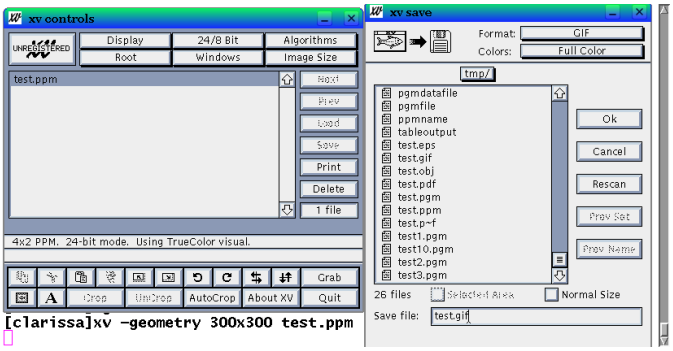


Figure 4.4: Controlling xv

figure of `xv` produces a window on the left handside of Fig. 4.4. When `save` button is clicked, another dialog window is produced which is the window on the right handside of Fig. 4.4. By selecting the `GIF` format, test.gif is saved in the current directory. This gif file can be imported into tgif. Fig. 4.5 shows the procedure to import this gif file `test.gif`.

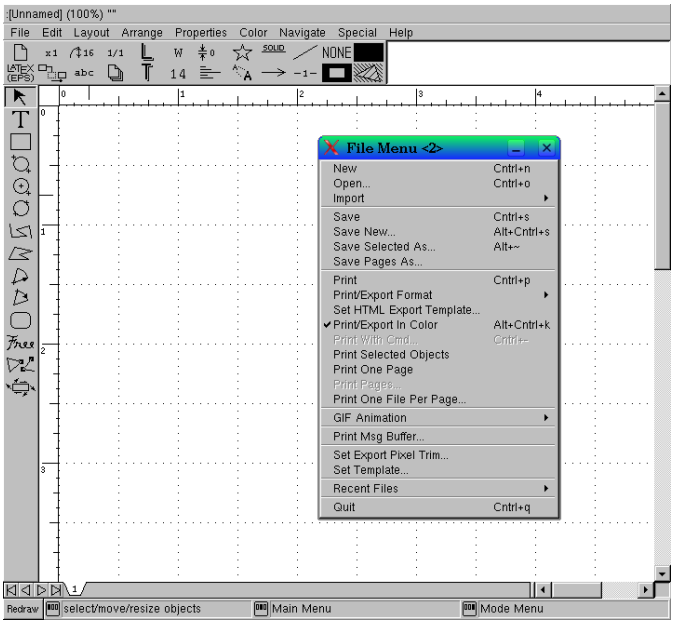


Figure 4.5: File importing in tgif

4.2 Assignment on 2D color presentation

Run `script`.

1. There are three files called `pgmdata1.gz`, `pgmdata2.gz` and `pgmdata3.gz`. Each datafile should be used to produce one pgm file. x-axis has 5 pixels and y-axis has 3 pixels.

   (a) Create a directory called `assignment4` under your home directory and move to the directory.

   (b) Unzip the datafiles `pgmdata?.gz` into the current directory by `cp /coursedisk/course/skeleton/pgmdata?.gz . ; gunzip pgmdata?.gz`

   (c) Modify the datafiles of `pgmdata?` as follows

   - `pgmdata1` should be multiplied by $2a$. Here a is the username number. If your username is eee98, this is achieved by
     ```
     cat pgmdata1 |
     awk '{print $1*98*2}'
     >  pgmdata1-new
     ```
     for example.
   - `pgmdata2` should be multiplied by $(3a + 1)$ If your username is eee98, this is achieved by
     ```
     cat pgmdata2 |
     awk '{print $1*(98*3+1)}'
     >  pgmdata2-new
     ```
     for example.
   - `pgmdata3` should be multiplied by $(4a + 2)$

   (d) Produce ascii pgm files from the modified pgmdata files and name them `test1.pgm`, `test2.pgm` and `test3.pgm`, respectively. While setting the maximum value which is fourth line in pgm file, try more than one value to find out the appropriate value for the fourth line(maximum value) in order to obtain the clear figure.

   (e) Convert pgm files to ppm files

   (f) Save these three figures in a JPEG format as `test1.jpg` , `test2.jpg` and `test3.jpg` ,respectively

   (g) Discuss your observation of `test?.jpg` in a file called `ppmobservation`.

   The assignment report should consist of the ascii file of `test?.pgm`, jpeg files of `test?.jpg` and `ppmobservation`.

2. There is a file called `/coursedisk/course/skeleton/whole.gz`. The file `whole.gz` has 165 columns in each line. If your username number is a, you use three columns of $3a - 2$, $3a - 1$, and $3a$ in the file `whole.gz`. If your username is eee31, this is achieved by
   ```
   cat whole  | awk '{print $90,$91,$92}'
   >  partwhole
   ```
   for example. The first column and the second column of `partwhole` show the x and y axis values, respectively. The third column of `partwhole` is the amplitude of E_z field at the point in the x-y plane.

   Plot the amplitude in the x-y plane with highest value in red and medium values in green and lowest value in purple and save the figure in pdf format. The ppm format may not be appropriate for some data due to the irregular spacing between points. Present the procedure in the report and save the figure in pdf format and name it `part.pdf`

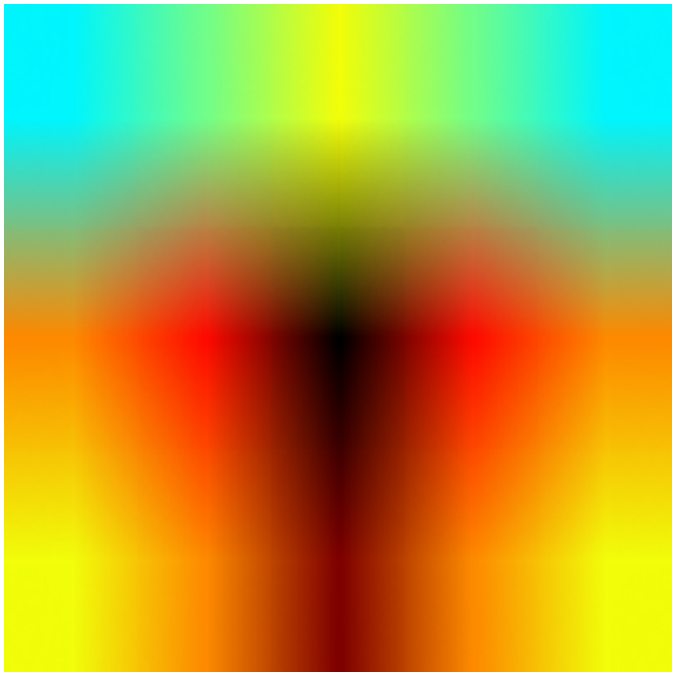


Figure 4.6: Example answer for test1.jpg

[1 mark]

Figure 4.7: Example answer for test2.jpg

[1 mark]

In assignment report, the properly modified data should be presented

[1 mark]

The discussion should state that

- pgmdata1 can be observed properly if the maximum value is set to $8\times(2a)$

[1 mark]

- pgmdata2 can be slightly observed by setting the real maximum value of $400\times(3a+1)$.

[1 mark]

If the maximum value is set to $308\times(3a+1)$, the figure passes the information which does not cover the entire data

[1 mark]

- pgmdata3 can not be viewed clearly because all the values are very similar relative to the maximum value even in color presentation.

[1 mark]

For the handling of the file whole, each user should have extracted the appropriate set of columns.

[0.5 marks]

One of the approaches to visualise the data in the file of whole is to use gnuplot like this explanation:

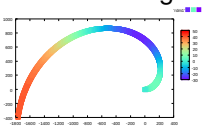
[2 marks]

- set pm3d
- set palette rgb 33,13,10
- set view map
- splot 'partwhole' w p palette pt 5 ps 1

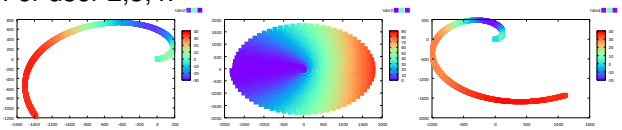
The saved pdf file should be close to the figures below depending on the usernumber:

[0.5 marks]

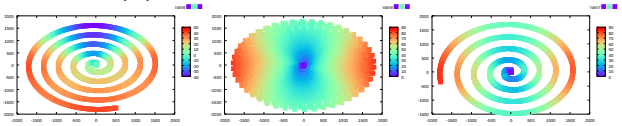
The following is the answer figures:For user 1:



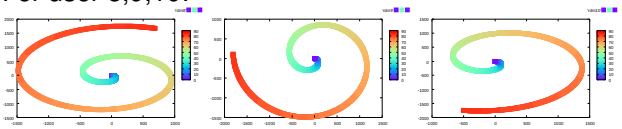
For user 2,3,4:



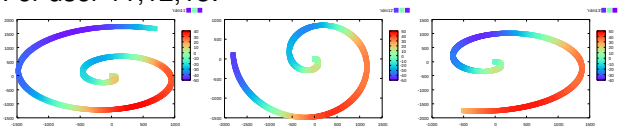
For user 5,6,7:



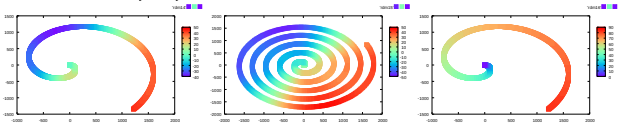
For user 8,9,10:



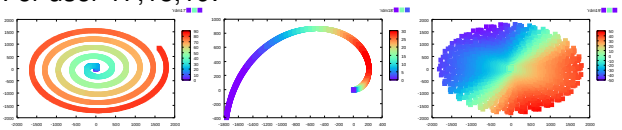
For user 11,12,13:



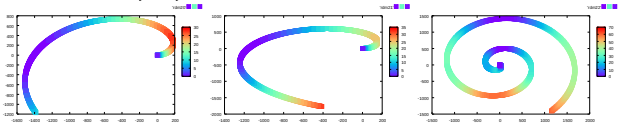
For user 14,15,16:



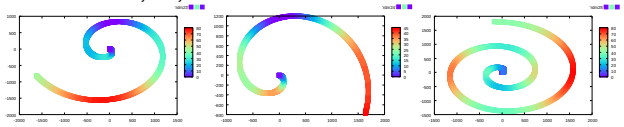
For user 17,18,19:



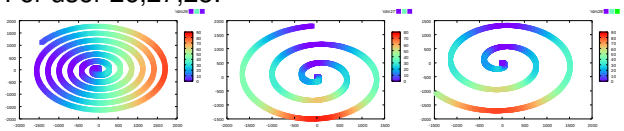
For user 20,21,22:



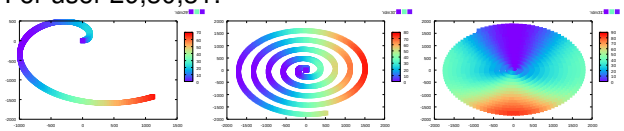
For user 23,24,25:



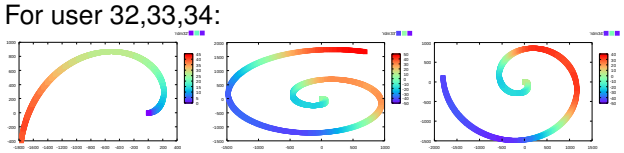
For user 26,27,28:



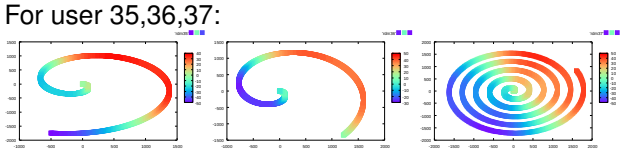
For user 29,30,31:



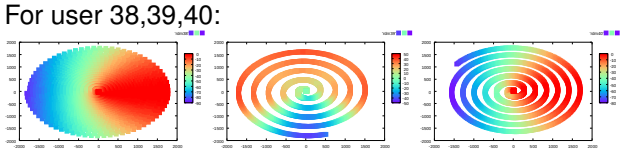
For user 32,33,34:



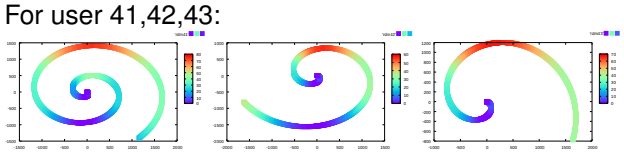
For user 35,36,37:



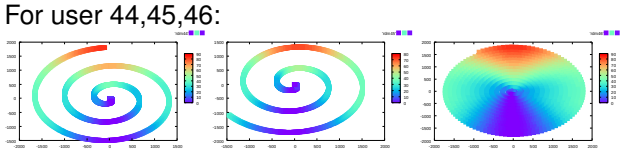
For user 38,39,40:



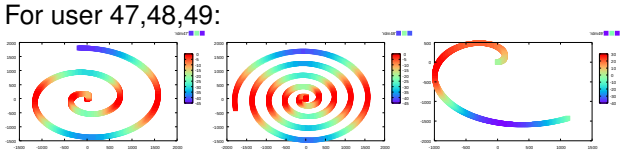
For user 41,42,43:



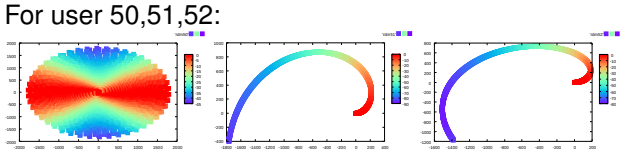
For user 44,45,46:



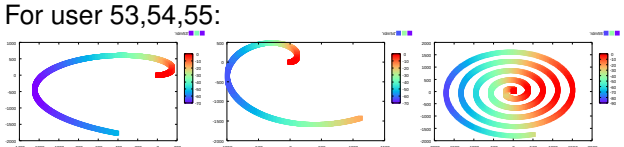
For user 47,48,49:



For user 50,51,52:



For user 53,54,55:



Submission of the assignment for EEEN40063

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Student signature	

.....

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Recipient signature	

Chapter 5

Signal processing to stress the characteristics of data

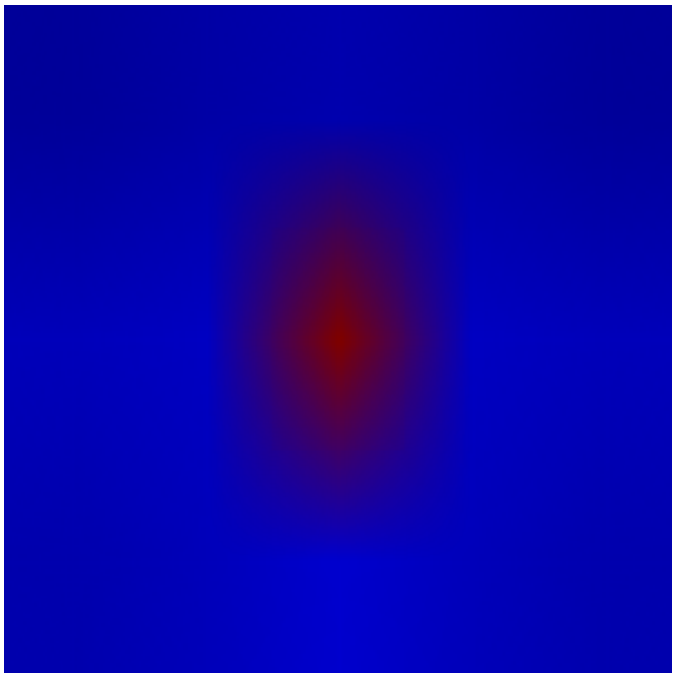


Figure 5.1: View via xv of pgmdata1— with the maximum value of 100

5.1 Alternative 2D contour for clearer view

When `pgmdata1` is converted into ppm format with the maximum value of 100, Fig. 5.1 is obtained. Fig. 5.1 does not provide enough precision for the values in the centre. `xv` plots data relative to the maximum value. The detail value range in the centre is 3∼8, whilst the maximum value is 100. Thus, the values around the centre area relative to the maximum value of 100 are indistinguishable. To solve this problem, high values should be suppressed by taking logarithm. This can be achieved by `cat test1.pgm | awk '(NR < 4){print $0}(NR > 3&&NR < 20) {print int(log(log($1)*10)*10)-23}' > test.pgm` which are all in one line. This means the following process:

1. Take logarithm of the value from the fourth line to the 19th line. At this point, the difference between minimum and maximum values becomes 3.5 but the values around the centre are indistinguishable as these are all 1.xxxxx and pgm file deals with only integer values.

2. Multiply the logarithm of the value by 10. At this point, the difference becomes about 35.

3. Repeat the same procedure to furthermore reduce the difference between the minimum value and the maximum value in the data. At this point, the difference becomes around 15. However, the minimum value of the data is about 63 % of the maximum value and it is expected to see a weak contrast in the figure.

4. Reduce all the values by 23 because the minimum value is 23. By this subtraction, the minimum value of the data becomes 0 and it is 0 % of the maximum value of 15. At this point, the high contrast can be obtained.

The resultant figure is presented in Fig. 5.2. When `pgmdata2` is converted into ppm format with the maximum value of 400, Fig. 5.3 is obtained. Without any modification, this color representation expresses the main characteristics of the data. When the details need to be presented, whole values have to be shifted toward zero and the difference between the minimum and the maximum should be reduced. One of the ways to achieve this is `cat test2.pgm | awk '(NR < 4){print $0}(NR > 3)&&(NR < 20) {print int(($1-200)**(0.9))}' > test.pgm` All the data is reduced by the minimum value of 200 and the reduced value to the power of 0.9 is shown in Fig. 5.4. When `pgmdata3` is converted into ppm format with the maximum value of 406, Fig. 5.5 is obtained. In Fig. 5.5, no pixels can be distinguished. This is because all the data is more than 99 % of the maximum value. To increase the ratio between the minimum value and the maximum value, data should be reduced by the current minimum value as is done for pgmdata1 and pgmdata2. `cat test3.pgm | awk '(NR < 4){print $0}(NR > 3)&&(NR < 20) {print $1-403}' > test.pgm` produces Fig. 5.6. When the location of maximum value needs to be stressed, the reduced data to the power of 2 ∼ 10 can be used. This is performed by `cat test3.pgm | awk '(NR < 4) {print $0}`

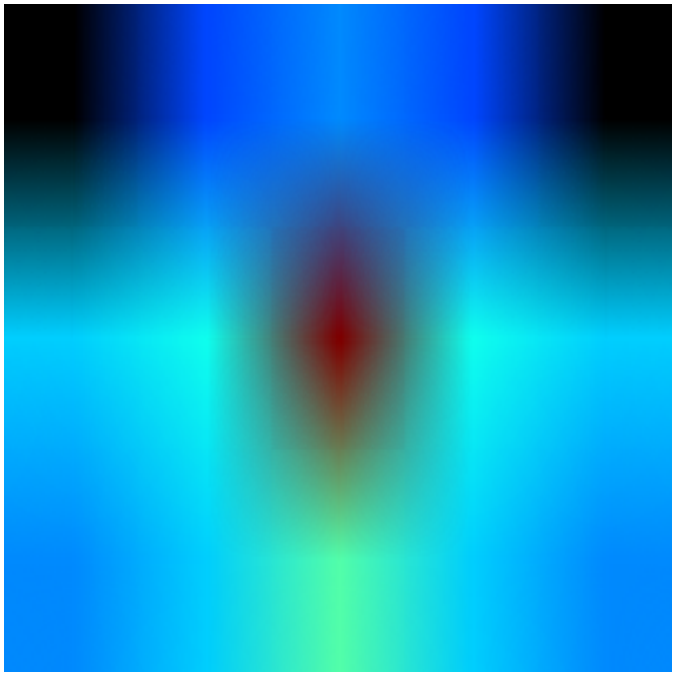


Figure 5.2: View via xv of pgmdata1 with the data processing as a ppm file

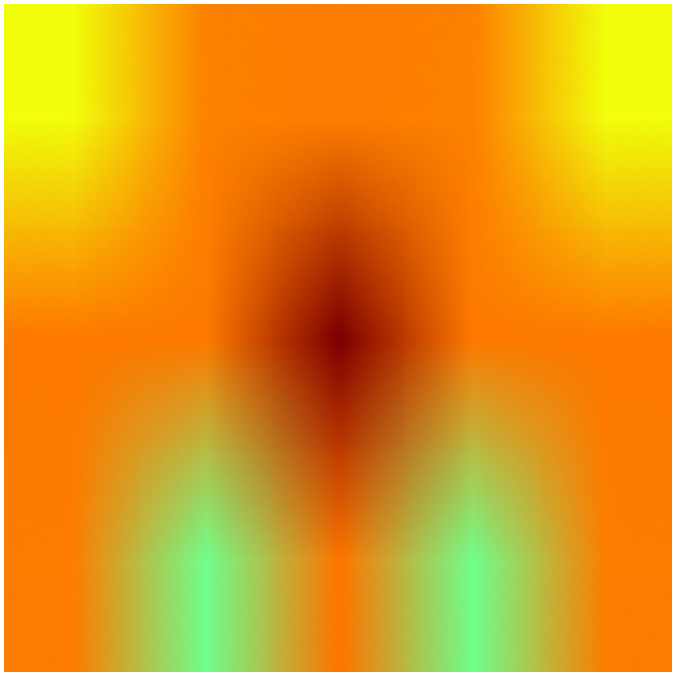


Figure 5.3: View via xv of pgmdata2 without the data processing as a ppm file

Figure 5.4: View via xv of pgmdata2 with the data processing as a ppm file

Figure 5.5: View via xv of pgmdata3 without the data processing as a ppm file

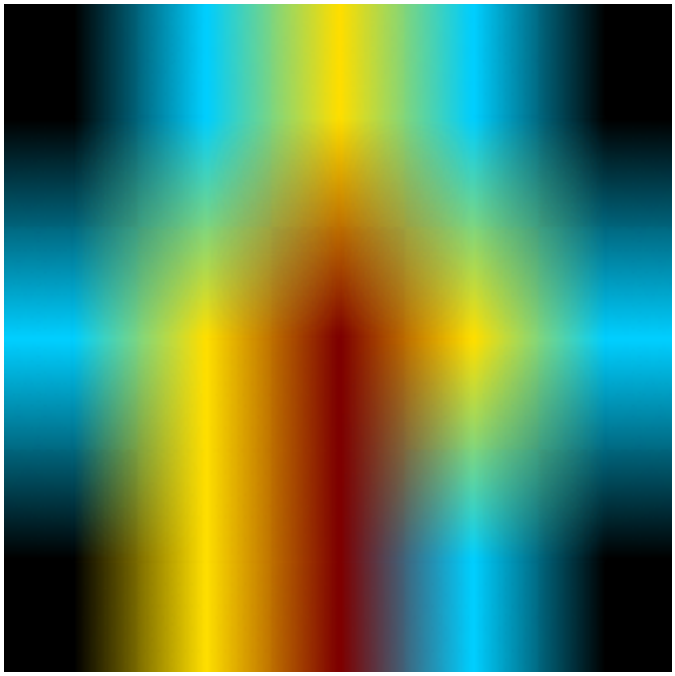


Figure 5.6: View via xv of pgmdata3 with the data processing as a ppm file

`(NR > 3)&&(NR < 20) {print ($1-403)**3}' > test.pgm` The resultant figure is Fig. 5.7.

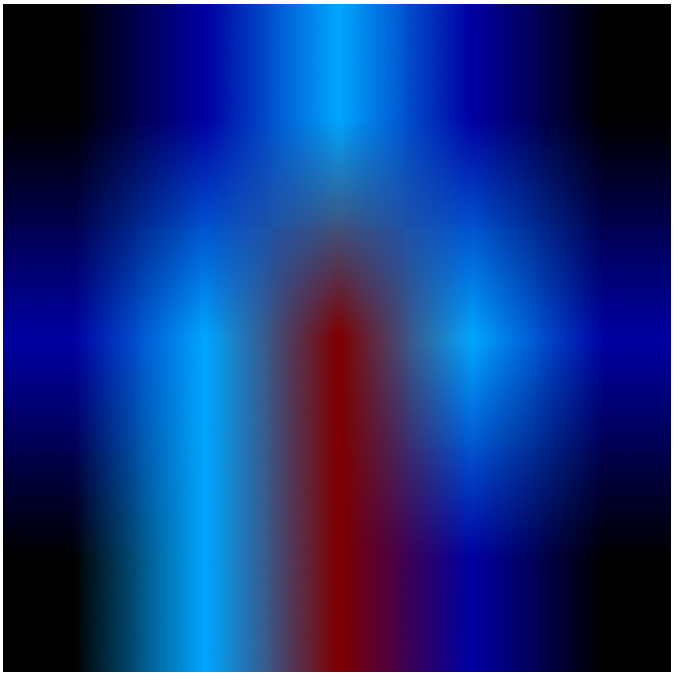


Figure 5.7: View via xv of pgmdata3 with the data processing to emphasise the location of the high values in the data as a ppm file

username number	column numbers	username number	column numbers	username number	column numbers	username number	column numbers
1	1,2,3	15	2,3,4	29	3,4,5	43	4,5,6
2	5,6,7	16	6,7,8	30	7,8,9	44	8,9,10
3	9,10,11	17	10,11,12	31	11,12,13	45	12,13,14
4	13,14,15	18	14,15,16	32	1,3,5	46	2,4,6
5	3,5,7	19	4,6,8	33	5,7,9	47	6,8,10
6	7,9,11	20	8,10,12	34	9,11,13	48	10,12,14
7	11,13,15	21	12,14,16	35	1,4,7	49	2,5,8
8	3,6,9	22	4,7,10	36	5,8,11	50	6,9,12
9	7,10,13	23	8,11,14	37	9,12,15	51	10,13,16
10	1,5,9	24	2,6,10	38	3,7,11	52	4,8,12
11	5,9,13	25	6,10,14	39	7,11,15	53	8,12,16
12	1,6,11	26	2,7,12	40	3,8,13	54	4,9,14
13	5,10,15	27	6,11,16	41	1,7,13	55	2,8,14
14	3,9,15	28	4,10,16	42	1,8,15	56	2,9,16

Table 5.1: Data allocation to students

5.2 Assignment on ppm for large data

`pgmdata4new.gz` has 16 columns and each column holds the data for 2 dimensional space with the size of 150×150.

1. Run a command `script` at the beginning of your work. This command will make a log of your usage of unix environment. The log is used for the part of the assessment.
2. Create a directory called `assignment5` under your home directory and switch to the directory. Copy and Unzip the datafile `/coursedisk/course/skeleton/pgmdata4new.gz` into the current directory.
3. Produce three ascii pgm files out of `pgmdata4new` and name them `test[1-16].pgm`, respectively. Find the three columns assigned to your username in Table 5.1.
4. Convert these as ppm format and name them `test[1-16].ppm`, respectively.
5. The procedure to produce the pgm format was not so straightforward; it can not be achieved by one command. Discuss which stage of the production should be simplified/automated in a file called `procedurediscussion`
6. Type `exit` at the end of your usage of the computer. This will produce a file called typescript. Change the name of the file to typescript-chap5 which should be in `/YourHomeDirectory/assignment5/`.

The unix commands such as sort, head and tail should be useful for your data processing and the explanation of these commands can be obtained by

```
man sort
man head
man tail
```

Your assignment report should contain the essence of the typescript file and the screen shot of xv view of three ppm files and the file `procedurediscussion`

```
[clarissa]echo  "P2"    >  test.pgm
[clarissa]echo "# test" >> test.pgm
[clarissa]echo "200 200" >> test.pgm
[clarissa]cat  pgmdata4 | awk '{print $2}' | sort -g | head -1
-64281058
[clarissa]cat pgmdata4 | awk '{print $2}' | sort -g | tail -1
64463355
[clarissa]echo "" | awk '{print 64463355+64281058}' >>test.pgm
[clarissa]cat pgmdata4 | awk '{print $2+64281058}' >> test.pgm
[clarissa]./pgm2ppm
[clarissa]cat test.ppm | awk '(NR == 1){print $1}(NR != 1){print $0}' > tmp
[clarissa]mv tmp test.ppm
[clarissa]
```

Figure 5.8: Example of log.pdf
The log should show

– the procedure to find out the maximum and minimum of the data

[2 marks]

– the procedure to convert the gray format to the color format

[1 mark]

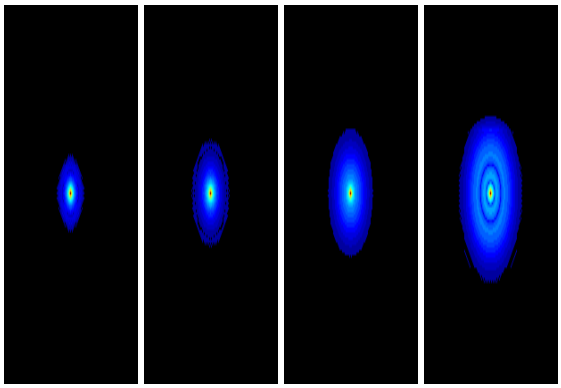


Figure 5.9: Example answer for ppm file for column1,2,3,4

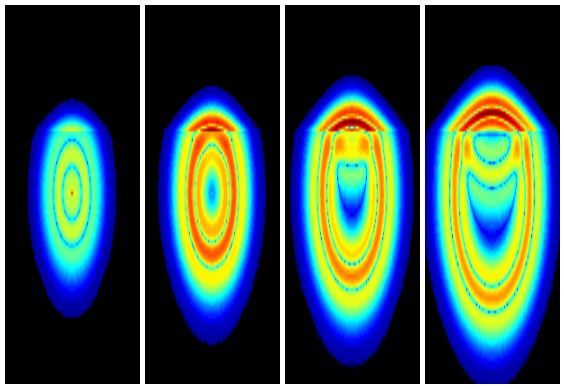


Figure 5.10: Example answer for ppm file for column5,6,7,8

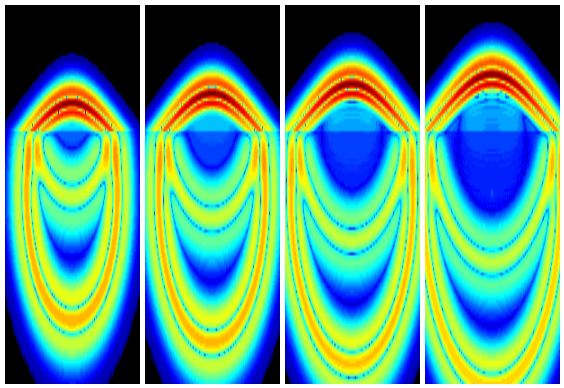


Figure 5.11: Example answer for ppm file for column9,10,11,12

For example, to deal with the second column, the Xwindow should show the command shown in Fig. 5.8
The discussion should say
the procedure to find the minimum and maximum value should be automated.

[2 marks]

Each student uses different columns from one another. The data used by student should match to Table 5.1

[2 marks]

Each student should produce three views among Fig. 5.9 $\sim$ Fig. 5.12. Each view should award 1 mark.

[3 marks]

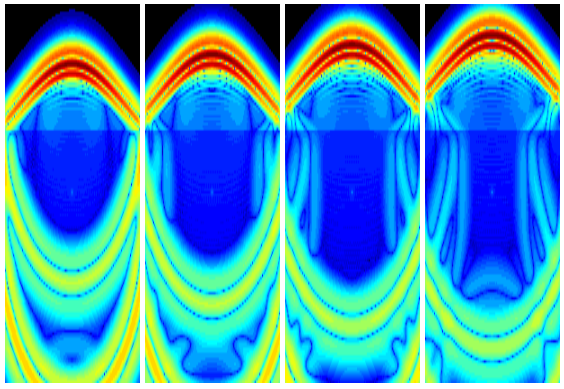


Figure 5.12: Example answer for ppm file for column13,14,15,16

Submission of the assignment for EEEN40063

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Student signature	

.....

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Recipient signature	

Chapter 6

Automated production of scientific data presentation

6.1 Shell script

Section 5.2 required several commands to produce a pgm file. However, when the number of required commands becomes large, it is easy to make a mistake at some point of the procedure. Unless the command `history` is used, it is difficult to track down all the commands you typed and to detect the command which is missed or mistaken. To avoid this problem and to automate the procedure, you can produce a shell script. The command lines you have typed so far can be put into a shell script and the shell executes all the commands in the scriptfile.

The ascii file should begin with

```
#!/bin/csh
```

for (t)csh and

```
#!/bin/bash
```

for bash. The command `echo $SHELL` will tell us which shell you are using. To work with tcsh, run `tcsh` because the default setting of your shell is bash. The following two shell scripts called `testshell` accomplish the same tasks but are written in a different shells.

```
#!/bin/csh
echo $argv
shift argv
shift argv
echo $argv
echo $#

if ($# < 3) then
echo "lack of input"
else
echo "Good"
endif


set a =  $argv[1]
set b =  $argv[2]
set c =  $argv[3]
@ d = $a + $b + $c
echo $d
pwd
set directoryname = `pwd`
echo "We are now "    $directoryname
set k = $argv[1]
while($k <  $argv[2] + 3 )
echo $k | awk -v s=4 '{print $1*s}'
@ k = $k + $argv[3]
end
```

```
 1  #!/bin/bash
 2  echo $@
 3
 4  shift 2
 5  echo $@
 6  echo $#
 7  if [ $# -lt 3 ]
 8  then {
 9  echo "lack of input"
10   }
11  else {
12  echo "Good"
13  }
14  fi
15  a=$1
16  b=$2
17  c=$3
18  d=$(($a+$b+$c))
19  echo $d
20  pwd
21  directoryname=`pwd`
22  echo "We are now "    $directoryname
23
24  for ((k=$1; $k< $2 + 3 ; k=$k + $3))
25  do
26  echo $k | gawk -v s=4  '{print $1*s}'
27  done
```

To run the shell script, the ordinary ascii file should be given the permission of the execution by

```
[clarissa] chmod +x testshell
```

For more information, type `man chmod`. Then, run the shell script:

```
[clarissa]./testshell 3 5 2 10 4
```

The shell script produces

```
3 5 2 10 4
2 10 4
3
Good
16
/home/fumie
We are now  /home/fumie
8
24
40
```

The content of the shell script can be understood as follows:

line 2 print out all the arguments

line 3-4 remove the first two arguments

line 5 print out all the arguments currently left

line 6 print out the number of all the arguments currently left

line 7-14 if the number of arguments is less than 3 print out `lack of input` and if the number of arguments is larger than 2 print out `Good`

line 15-17 save the value of first three arguments 1∼3 into variables `a` ∼ `c`.

line 18-19 add these three arguments and print out

line 20 execute `pwd`

line 21-22 save the result of `pwd` into a variable of `directoryname` and print it out

line 23-27 set a variable `k` to the value of the first argument and repeat the following till `k` reaches the summation of the second argument and 3.

line 25 print out the multiplication of value `s` which is 4 and `k`

line 26 `k` increases by the value of the third argument

6.2 Assignment on shell script for 3D integer data

`http://www.linuxconfig.org/Bash_scripting_Tutorial` is one of the source for bash-scripting.

FDTD space of the size $150 \times 150 \times 150$ is excited by a dipole antenna oriented in the z direction at the location (75,75,75).
The parameters of $\mathcal{X} = 300$, $f_{CW} =$ 3GHz. $w = 5$ and $\alpha = 2.3$ are used for the FDTD computation which you do not have to understand now. `data-a-b-b-0-(x,z-)simple.tar.gz` has `3ddatasimple???` which have 4 columns.

`data-a-b-b-0-simple.tar.gz`
has FDTD space filled with the conductivity of a [S/m] and relative permittivity of b.

`data-a-b-b-0-x-simple.tar.gz`
has FDTD space filled with air apart from the space with $1 \leq x \leq 25$ whose medium parameters are the conductivity of a[S/m] and relative permittivity of b.

`data-a-b-b-0-z-simple.tar.gz`
has FDTD space filled with air apart from the space with $1 \leq z \leq 25$ whose medium parameters are the conductivity of a[S/m] and relative permittivity of b.

Here`???` is $50n$ where n = 1, 2, $\cdots$, 10. First, second and third columns are x, y and z values, respectively and fourth column has $\mathbf{E}$ field value. $\mathbf{E}$ on $z = 75$ plane or $x = 75$ plane need to be plotted.

- Create a directory called `assignment6` under your home directory and switch to the directory.
- Untar the datafile `data-a-b-b-0-simple.tar.gz` into the current directory by `tar -zxvf /coursedisk/course/skeleton/data-a-b-b-0-simple.tar.gz`. Table 6.1 shows the name of the data file each user is supposed to use and the target plane to view.

Table 6.1: Data file and view plane allocation

user number	file name	plane to see
1	data-1000-1-1-0-x-simple.tar.gz	z
2	data-0-1-1-0-simple.tar.gz	x
3	data-1000-1-1-0-simple.tar.gz	z
4	data-0-10-10-0-simple.tar.gz	x
5	data-100-1-1-0-z-simple.tar.gz	z
6	data-0-10-10-0-x-simple.tar.gz	x
7	data-100-1-1-0-x-simple.tar.gz	z
8	data-0-10-10-0-z-simple.tar.gz	x
9	data-100-1-1-0-simple.tar.gz	z
10	data-0-2-2-0-simple.tar.gz	x
11	data-10-1-1-0-z-simple.tar.gz	z
12	data-0-2-2-0-x-simple.tar.gz	x
13	data-10-1-1-0-x-simple.tar.gz	z
14	data-0-2-2-0-z-simple.tar.gz	x
15	data-10-1-1-0-simple.tar.gz	z
16	data-0-20-20-0-simple.tar.gz	x
17	data-1-1-1-0-z-simple.tar.gz	z
18	data-0-20-20-0-x-simple.tar.gz	x
19	data-1-1-1-0-x-simple.tar.gz	z
20	data-0-20-20-0-z-simple.tar.gz	x
21	data-1-1-1-0-simple.tar.gz	z
22	data-0-30-30-0-simple.tar.gz	x
23	data-0.1-1-1-0-z-simple.tar.gz	z
24	data-0-30-30-0-x-simple.tar.gz	x
25	data-0.1-1-1-0-x-simple.tar.gz	z
26	data-0-30-30-0-z-simple.tar.gz	x
27	data-0.1-1-1-0-simple.tar.gz	z
28	data-0-5-5-0-simple.tar.gz	x
29	data-0-5-5-0-z-simple.tar.gz	z
30	data-0-5-5-0-x-simple.tar.gz	x

user number	file name	plane to see
31	data-0-5-5-0-x-simple.tar.gz	z
32	data-0-5-5-0-z-simple.tar.gz	x
33	data-0-5-5-0-simple.tar.gz	z
34	data-0.1-1-1-0-simple.tar.gz	x
35	data-0-30-30-0-z-simple.tar.gz	z
36	data-0.1-1-1-0-x-simple.tar.gz	x
37	data-0-30-30-0-x-simple.tar.gz	z
38	data-0.1-1-1-0-z-simple.tar.gz	x
39	data-0-30-30-0-simple.tar.gz	z
40	data-1-1-1-0-simple.tar.gz	x
41	data-0-20-20-0-z-simple.tar.gz	z
42	data-1-1-1-0-x-simple.tar.gz	x
43	data-0-20-20-0-x-simple.tar.gz	z
44	data-1-1-1-0-z-simple.tar.gz	x
45	data-0-20-20-0-simple.tar.gz	z
46	data-10-1-1-0-simple.tar.gz	x
47	data-0-2-2-0-z-simple.tar.gz	z
48	data-10-1-1-0-x-simple.tar.gz	x
49	data-0-2-2-0-x-simple.tar.gz	z
50	data-10-1-1-0-z-simple.tar.gz	x
51	data-0-2-2-0-simple.tar.gz	z
52	data-100-1-1-0-simple.tar.gz	x
53	data-0-10-10-0-z-simple.tar.gz	z
54	data-100-1-1-0-x-simple.tar.gz	x
55	data-0-10-10-0-x-simple.tar.gz	z
56	data-100-1-1-0-z-simple.tar.gz	x
57	data-0-10-10-0-simple.tar.gz	z
58	data-1000-1-1-0-simple.tar.gz	x
59	data-0-1-1-0-simple.tar.gz	z
60	data-1000-1-1-0-x-simple.tar.gz	x

For example, user with number 8 should use the data file `data-0-10-10-0-z-simple.tar.gz` and $x = 75$ plane should be visualised.

- Write a shell script named `3dsimpleshell` which
  - produces and saves ascii pgm files out of `3ddatasimple???`
  - converts these as ppm format
- Save the image of ppm file in pdf format and name it `3ddatasimple???.pdf`.

Assignment report should contain `3dsimpleshell` and explanation of the function of each line of `3dsimpleshell` and `3ddatasimple???.pdf`.

```
[clarissa]cat -n demosimpleshell
     1  #!/bin/csh
     2  set fixlocation = 75
     3  set plaintosee = 3
     4  set seecolumn = 4
     5  set xaxis = 1
     6  set yaxis = 2
     7  set k = 100
     8  while ($k <  210)
     9  set maxvalue = 1000
    10  cat 3ddatasimple$k | gawk -v fixlocation=$fixlocation -v plaintosee=$plaintosee -v seecolum
n=$seecolumn -v xaxis=$xaxis -v yaxis=$yaxis '($plaintosee == fixlocation){print $xaxis,$yaxis,(($s
eecolumn*$seecolumn)) }'>test
    11  set xrangemax =  `cat test | sort -n -k 1 | tail -1 | awk  '{print $1}'`
    12  set xrangemin =  `cat test | sort -nr -k 1 | tail -1 | awk  '{print $1}'`
    13  @ xrange =  1 + $xrangemax - $xrangemin
    14  set yrangemax =  `cat test | sort -n -k 2  | tail -1 | awk  '{print $2}'`
    15  set yrangemin =  `cat test | sort -nr -k 2 | tail -1 | awk  '{print $2}'`
    16  @ yrange =  1 + $yrangemax - $yrangemin
    17  echo "P2" > test.pgm
    18  echo "# test" >> test.pgm
    19  echo $xrange $yrange  >> test.pgm
    20  cat 3ddatasimple$k | sort -gr -k 4 | head -1 | awk '{print $4}' >> test.pgm
    21  cat 3ddatasimple$k | awk '{print $4}' >> test.pgm
    22  cat test.pgm | awk '((NR > 4)&&($1 < 1)){print 0 }((NR > 4)&& ($1 > 0)){print $0}(NR < 5){p
rint $0}' > testtest.pgm
    23  mv testtest.pgm test.pgm
    24  /home/fumie/latex/modules/teachingdata/pgm2ppm
    25  cat test.ppm | awk '(NR == 1){print $1}(NR > 1){print $0}' > test$k.ppm
    26  @ k = $k + 10
    27  end
    28  set files = ""
    29  set k = 100
    30  while($k < 210)
    31  set files = "$files test$k.ppm"
    32  @ k = $k + 10
    33  end
    34  xv -wait 0.0001 $files
    35  exit 0
[clarissa]
```

Figure 6.1: Example answer for 3dsimpleshell

– Finding out the maximum value in the data for the fourth line in pgm file (line 20)

[2 marks]

– Producing pgm file (line 17-21)

[2 marks]

– Converting pgm file to ppm file(line 24-25)

[1 mark]

The shell script should include most of Fig. 6.1 EXCEPT the display with xv. There is no need to display ppm files at the moment. 3dsimpleshell of each user should differ from Fig. 6.1 on the points indicated in Table 6.2. The view plane in Fig. 6.1 is $z = 75$. This has to be modified by students who need to see $x = 75$ plane. These modification should be presented in students' 3dsimpleshell.

[3 marks]

3ddatasimple???.pdf which each student produces should be identical to the figures in /local/fumietest-a-b-b-0-x-movie/x75, /local/fumietest-a-b-b-0-x-movie/z75, /local/fumietest-a-b-b-0-z-movie/x75, /local/fumietest-a-b-b-0-z-movie/z75, /local/fumietest-a-b-b-0-movie/x75, /local/fumietest-a-b-b-0-movie/z75, on 130.88.154.35.

[2 marks]

Table 6.2: Required modification of Fig. 6.1 for each student's 3dsimpleshell

userNo	file name	set k = ??	k = $k + ??
1	1000-1-1-0-x	130	40
2	0-1-1-0	101	11
3	1000-1-1-0	129	39
4	0-10-10-0	102	12
5	100-1-1-0-z	128	38
6	0-10-10-0-x	103	13
7	100-1-1-0-x	127	37
8	0-10-10-0-z	104	14
9	100-1-1-0	126	36
10	0-2-2-0	105	15
11	10-1-1-0-z	125	35
12	0-2-2-0-x	106	16
13	10-1-1-0-x	124	34
14	0-2-2-0-z	107	17
15	10-1-1-0	123	33
16	0-20-20-0	108	18
17	1-1-1-0-z	122	32
18	0-20-20-0-x	109	19
19	1-1-1-0-x	121	31
20	0-20-20-0-z	110	20
21	1-1-1-0	120	30
22	0-30-30-0	111	21
23	0.1-1-1-0-z	119	29
24	0-30-30-0-x	112	22
25	0.1-1-1-0-x	118	28
26	0-30-30-0-z	113	23
27	0.1-1-1-0	117	27
28	0-5-5-0	114	24
29	0-5-5-0-z	116	26
30	0-5-5-0-x	115	25
userNo	file name	set k = ??	k = $k + ??
31	0-5-5-0-x	115	25
32	0-5-5-0-z	116	26
33	0-5-5-0	114	24
34	0.1-1-1-0	117	27
35	0-30-30-0-z	113	23
36	0.1-1-1-0-x	118	28
37	0-30-30-0-x	112	22
38	0.1-1-1-0-z	119	29
39	0-30-30-0	111	21
40	1-1-1-0	120	30
41	0-20-20-0-z	110	20
42	1-1-1-0-x	121	31
43	0-20-20-0-x	109	19
44	1-1-1-0-z	122	32
45	0-20-20-0	108	18
46	10-1-1-0	123	33
47	0-2-2-0-z	107	17
48	10-1-1-0-x	124	34
49	0-2-2-0-x	106	16
50	10-1-1-0-z	125	35
51	0-2-2-0	105	15
52	100-1-1-0	126	36
53	0-10-10-0-z	104	14
54	100-1-1-0-x	127	37
55	0-10-10-0-x	103	13
56	100-1-1-0-z	128	38
57	0-10-10-0	102	12
58	1000-1-1-0	129	39
59	0-1-1-0	101	11
60	1000-1-1-0-x	130	40

Submission of the assignment for EEEN40063

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Student signature	

.....

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Recipient signature	

Chapter 7

Extraction of the time domain signal from the spatial domain signal

7.1 FDTD output

FDTD calculates and records the field distribution at each time step in FDTD space. File recorded at each time step has four columns:

First column value of x-axis of a certain point in FDTD space

Second column value of y-axis of a certain point in FDTD space

Third column value of z-axis of a certain point in FDTD space

Fourth column value of $\mathbf{E}$ at a certain point in FDTD space

In previous chapters, the field distribution was dealt with on the slice of the FDTD space, for example on $z = 75\ \Delta s$ plane. To extract the time domain signal at a certain spatial point, for example, $x = 12$,$y = 45$ and $z = 75$, a signal at this spatial point has to be collected from each time step FDTD output. This is achieved by running the following command; `gawk '($1== 12 && $2 == 45 && $3 == 75) {print ARGIND, $4}' E*out` This means if the first column of the datafile `E*out` is 12 and the second column of the datafile `E*out` is 45 and the third column of the datafile `E*out` is 75, then print out the fourth column. In this module, the FDTD calculation is performed in the uniform spatial discretisation ($\Delta x = \Delta y = \Delta z \stackrel{\triangle}{=} \Delta s$) using the stability condition in Eq. 7.1 as the equality.

$$v\Delta t \leq (\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2})^{-\frac{1}{2}} \tag{7.1}$$

When $\Delta s = \frac{\lambda}{\mathcal{X}}$ and $\lambda = \frac{v}{f}$, $\Delta t = \frac{\Delta s}{\sqrt{3}v} = \frac{\lambda}{\sqrt{3}v\mathcal{X}} = \frac{1}{\sqrt{3}\mathcal{X}f}$. In this module, f is set to 3GHz.

`gawk '($1 == 12 && $2 == 45 && $3 == 75) {print ARGIND, $4}' E*out > plot.data` will produce `plot.data`. The first column of `plot.data` is a series of integers $1, 2, 3, \cdots$. and the second column is the value of $\mathbf{E}$ over the time. If the first column is 5, the second column means $\mathbf{E}$ value at (12,45,75) in FDTD space at 5 Δt. When $\mathcal{X} = 140$, the file `plot.data` should be plotted in `gnuplot` as

```
plot 'plot.data'
 u ($1/sqrt(3)/140/3e9):2 w l
```

so that the x -axis can be plotted in second. In scientific presentation, x- and y- axises has to be plotted with an appropriate unit. This could be tried out using `PEC.tar.gz`.

As is shown, `xv` displays ppm files. `xv -wait 5 a.ppm` will show `a.ppm` for 5 second and terminate the display automatically. If `xv -wait 0.1 a.ppm ; xv -wait 0.1 b.ppm` is performed, the termination of `a.ppm` is observed before `b.ppm` is seen. When the smooth visualisation is desired, `xv -wait 0.01 a.ppm b.ppm` should be performed. Here, the number after `-wait` can have any number such as 10 or 0.0001. There is no termination of window between the display of `a.ppm` and `b.ppm`.

7.2 Assignment on shell script for 3D real data

FDTD space whose size is $150 \times 150 \times 150$ is excited at the centre as is the case with Section 6.2. $\mathbf{E}$ distribution from FDTD calculation are in `3ddata???` . ??? is the number of the time steps from the starting point of the calculation. First, second and third columns are x, y and z values, respectively and fourth column has $\mathbf{E}$ field value. The power distribution ($|\mathbf{E}|^2$) on $z = 75$ or $x = 75$ needs to be plotted.

- Create a directory called `assignment7` under your home directory and move to the directory.
- Untar the data file into the current directory. Table 7.1 shows the name of the data file each user is supposed to use and the view plane. For example, if your user number is 15, use `data-0-10-10-0-x.tar.gz` and visualise $z = 75$ plane. It is important that you take the right dataset so that proper assessment can be performed because each dataset has different information.
- Write a shell script named `3dshell` which
  - produces and saves ascii pgm files out of `3ddata???`
  - converts these as ppm format such as `test123.ppm`.
  - displays these ppm file successively using `xv`.

The assignment report should contain `3dshell` and your idea on designing the shell script. Please don't delete your shell script and both pgm and ppm files you produced in your working directory. These are assessed against the individual predefined answers. Launch `script` before starting your work. Accumulate the typescript in a file in your working directory.

Table 7.1: Data file and view plane allocation

user number	file name	plane to see
1	data-0-20-20-0-z.tar.gz	z
2	data-1-1-1-0-x.tar.gz	x
3	data-0-20-20-0-x.tar.gz	z
4	data-1-1-1-0-z.tar.gz	x
5	data-0-20-20-0.tar.gz	z
6	data-10-1-1-0.tar.gz	x
7	data-0-2-2-0-z.tar.gz	z
8	data-10-1-1-0-x.tar.gz	x
9	data-0-2-2-0-x.tar.gz	z
10	data-10-1-1-0-z.tar.gz	x
11	data-0-2-2-0.tar.gz	z
12	data-100-1-1-0.tar.gz	x
13	data-0-10-10-0-z.tar.gz	z
14	data-100-1-1-0-x.tar.gz	x
15	data-0-10-10-0-x.tar.gz	z
16	data-100-1-1-0-z.tar.gz	x
17	data-0-10-10-0.tar.gz	z
18	data-1000-1-1-0.tar.gz	x
19	data-0-1-1-0.tar.gz	z
20	data-1000-1-1-0-x.tar.gz	x
21	data-1000-1-1-0-x.tar.gz	z
22	data-0-1-1-0.tar.gz	x
23	data-1000-1-1-0.tar.gz	z
24	data-0-10-10-0.tar.gz	x
25	data-100-1-1-0-z.tar.gz	z
26	data-0-10-10-0-x.tar.gz	x
27	data-100-1-1-0-x.tar.gz	z
28	data-0-10-10-0-z.tar.gz	x
29	data-100-1-1-0.tar.gz	z
30	data-0-2-2-0.tar.gz	x

user number	file name	plane to see
31	data-10-1-1-0-z.tar.gz	z
32	data-0-2-2-0-x.tar.gz	x
33	data-10-1-1-0-x.tar.gz	z
34	data-0-2-2-0-z.tar.gz	x
35	data-10-1-1-0.tar.gz	z
36	data-0-20-20-0.tar.gz	x
37	data-1-1-1-0-z.tar.gz	z
38	data-0-20-20-0-x.tar.gz	x
39	data-1-1-1-0-x.tar.gz	z
40	data-0-20-20-0-z.tar.gz	x
41	data-1-1-1-0.tar.gz	z
42	data-0-30-30-0.tar.gz	x
43	data-0.1-1-1-0-z.tar.gz	z
44	data-0-30-30-0-x.tar.gz	x
45	data-0.1-1-1-0-x.tar.gz	z
46	data-0-30-30-0-z.tar.gz	x
47	data-0.1-1-1-0.tar.gz	z
48	data-0-5-5-0.tar.gz	x
49	data-0-5-5-0-z.tar.gz	z
50	data-0-5-5-0-x.tar.gz	x
51	data-0-5-5-0-x.tar.gz	z
52	data-0-5-5-0-z.tar.gz	x
53	data-0-5-5-0.tar.gz	z
54	data-0.1-1-1-0.tar.gz	x
55	data-0-30-30-0-z.tar.gz	z
56	data-0.1-1-1-0-x.tar.gz	x
57	data-0-30-30-0-x.tar.gz	z
58	data-0.1-1-1-0-z.tar.gz	x
59	data-0-30-30-0.tar.gz	z
60	data-1-1-1-0.tar.gz	x

```
[clarissa]cat -n demoshell
     1  #!/bin/csh
     2  set fixlocation = 75
     3  set plaintosee = 3
     4  set seecolumn = 4
     5  set xaxis = 1
     6  set yaxis = 2
     7  set k = 100
     8  while ($k <  210)
     9  set maxvalue = 1000
    10  cat 3ddata$k | gawk -v fixlocation=$fixlocation -v plaintosee=$plaintosee -v seecolumn=$seecolumn -v xa
xis=$xaxis -v yaxis=$yaxis '($plaintosee == fixlocation){print $xaxis,$yaxis,(($seecolumn*$seecolumn)) }'>test
    11  set xrangemax =  `cat test | sort -n -k 1 | tail -1 | awk  '{print $1}'`
    12  set xrangemin =  `cat test | sort -nr -k 1 | tail -1 | awk  '{print $1}'`
    13  @ xrange =  1 + $xrangemax - $xrangemin
    14  set yrangemax =  `cat test | sort -n -k 2  | tail -1 | awk  '{print $2}'`
    15  set yrangemin =  `cat test | sort -nr -k 2 | tail -1 | awk  '{print $2}'`
    16  @ yrange =  1 + $yrangemax - $yrangemin
    17  set min = `cat test | sort -gr -k 3 | tail -1 | awk '{print $3}'`
    18  set max = `cat test | sort -g -k 3 | tail -1 | awk '{print $3}'`
    19  set range = `echo ""|awk -v min=$min -v max=$max '{print max - min}'`
    20  cat test | awk -v range=$range -v min=$min -v maxvalue=$maxvalue '{print $1,$2,int((($3-min)/range)**0
.2*maxvalue)**0.5)}' > test2
    21  echo "P2" > test.pgm
    22  echo "# test" >> test.pgm
    23  echo $xrange $yrange  >> test.pgm
    24  echo ""|  awk -v maxvalue=$maxvalue '{print  int((maxvalue)**0.5)}'  >> test.pgm
    25  cat test2 | awk '{print $3}' >> test.pgm
    26  cat test.pgm | awk '((NR > 4)&&($1 < 1)){print 0 }((NR > 4)&& ($1 > 0)){print $0}(NR < 5){print $0}' >
testtest.pgm
    27  mv testtest.pgm test.pgm
    28  /home/fumie/latex/modules/teachingdata/pgm2ppm
    29  cat test.ppm | awk '(NR == 1){print $1}(NR > 1){print $0}' > test$k.ppm
    30  @ k = $k + 10
    31  end
    32  set files = ""
    33  set k = 100
    34  while($k < 210)
    35  set files = "$files test$k.ppm"
    36  @ k = $k + 10
    37  end
    38  xv -wait 0.0001 $files
    39  exit 0
[clarissa]
```

Figure 7.1: Example answer for 3dshell

– production of pgm file

[2 marks]

– production of ppm file

[1 mark]

– visualisation in xv

[2 marks]

The shell script should include most of Fig. 7.1. However, Table 7.2 (difference from Fig. 7.1, depending on the username number) should be observed in 3dshell

[2 marks]

The correct shell scripts and resultant ppm files are `130.88.154.35:/local/fumietest-*-movie`.
Ppm files which are produced by students should be checked electrically

[3 marks]

Table 7.2: The expected difference between Fig. 7.1 and 3dshell which each student produces

userNo.	data file	set k =	$k increment
1	data-0-20-20-0-x	152	52
2	data-1-1-1-0	130	30
3	data-0-20-20-0	154	54
4	data-1-1-1-0-x	128	28
5	data-0-20-20-0-z	150	50
6	data-10-1-1-0-z	120	20
7	data-0-2-2-0-x	158	58
8	data-10-1-1-0	124	24
9	data-0-2-2-0	160	60
10	data-10-1-1-0-x	122	22
11	data-0-2-2-0-z	156	56
12	data-100-1-1-0-z	114	14
13	data-0-10-10-0-x	164	64
14	data-100-1-1-0	118	18
15	data-0-10-10-0	166	66
16	data-100-1-1-0-x	116	16
17	data-0-10-10-0-z	162	62
18	data-1000-1-1-0-x	110	10
19	data-0-1-1-0	168	68
20	data-1000-1-1-0	112	12
21	data-1000-1-1-0	112	12
22	data-0-1-1-0	168	68
23	data-1000-1-1-0-x	110	10
24	data-0-10-10-0-z	162	62
25	data-100-1-1-0-x	116	16
26	data-0-10-10-0	166	66
27	data-100-1-1-0	118	18
28	data-0-10-10-0-x	164	64
29	data-100-1-1-0-z	114	14
30	data-0-2-2-0-z	156	56
userNo.	data file	set k =	$k increment
31	data-10-1-1-0-x	122	22
32	data-0-2-2-0	160	60
33	data-10-1-1-0	124	24
34	data-0-2-2-0-x	158	58
35	data-10-1-1-0-z	120	20
36	data-0-20-20-0-z	150	50
37	data-1-1-1-0-x	128	28
38	data-0-20-20-0	154	54
39	data-1-1-1-0	130	30
40	data-0-20-20-0-x	152	52
41	data-1-1-1-0-z	126	26
42	data-0-30-30-0-z	144	44
43	data-0.1-1-1-0-x	134	34
44	data-0-30-30-0	148	48
45	data-0.1-1-1-0	136	36
46	data-0-30-30-0-x	146	46
47	data-0.1-1-1-0-z	132	32
48	data-0-5-5-0-z	138	38
49	data-0-5-5-0-x	140	40
50	data-0-5-5-0	142	42
51	data-0-5-5-0	142	42
52	data-0-5-5-0-x	140	40
53	data-0-5-5-0-z	138	38
54	data-0.1-1-1-0-z	132	32
55	data-0-30-30-0-x	146	46
56	data-0.1-1-1-0	136	36
57	data-0-30-30-0	148	48
58	data-0.1-1-1-0-x	134	34
59	data-0-30-30-0-z	144	44
60	data-1-1-1-0-z	126	26

Submission of the assignment for EEEN40063

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Student signature	

.....

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Recipient signature	

Chapter 8

Frequency domain signals

8.1 Discrete Fourier Transform

Discrete Fourier Transform converts a time domain signal to a frequency domain signal. An in-house program `/coursedisk/course/skeleton/calspectrum` takes two files as input files. One file is called `timedata`. This file has one column time domain data. The other file is called `initialdata` The format of `initialdata` is

```
multiplifier, 1
element_per_wavelength,     150
wave_frequency,   3e9
maxsteps,     5000
```

Second row of `initialdata` has the information on $\mathcal{X}$ which is used for the calculation of FDTD to set the spatial resolution of $0.1/\mathcal{X}$ meters. Third row is the nominal frequency which is used for the calculation of Δs in FDTD. Fourth row is length of the time domain signal `timedata`. If `timedata` is given, the length of the data can be worked out by `wc -l timedata`. If a part of the signal from the datafile of `plot.data` is extracted by, say, `head -5678 plot.data | tail -3456 | awk '{print $2} ' > timedata`, 3456 is supposed to be written as a replacement of `5000` in the fourth line of `initialdata` This number of data `maxsteps` is noted as N. If you want to skim the original data by `cat plot.data | awk '(NR % 3 == 1){print $2}' > timedata` which prints out every three data, the first line should be `multiplifier, 3`.
The program `calspectrum` calculates the temporal discretisation based on $\Delta t = \frac{1}{\sqrt{3} f \mathcal{X}}$ and converts this time domain signal to a frequency domain signal. The signal in frequency domain is complex and `calspectrum` produces two files; `outputreal` and `outputimag`. `outputreal` has two columns. The first column is the frequency. The frequency increases regularly with the increase of a row. The frequency spacing is noted as Δf. Δf satisfies $\Delta f \times \Delta t \times N = 1$. The second column is the real part of the frequency spectrum at a certain frequency. `outputimag` is very similar to `outputreal`. The difference between these two files is that `outputimag` prints out the imaginary part of the spectrum, not real part. The spectrum magnitude can be produced by `paste outputreal outputimag | awk '{print $1, $2**2+$4**2}' > output`

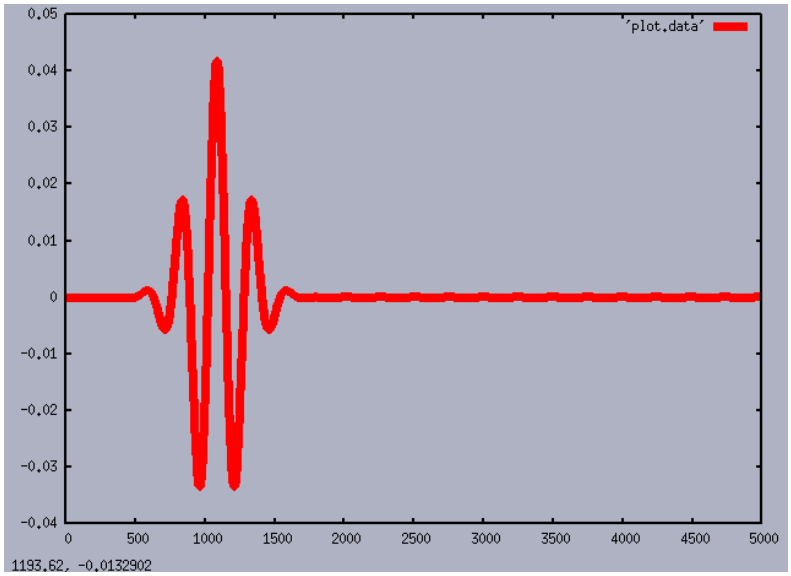


Figure 8.1: Example of a time domain signal with direct signal and resonant signal

The dataset is in `PEC.tar.gz` .

For example, a time domain signal is observed at (48,50,50) in Fig. 8.1 which is produced by gnuplot with the command of `plot 'plot.data' w l lw 7` . The signal up to 1500 time steps shows the waveform directly propagated from the source excitation location. When this time domain signal upto 2000 time steps is turned into the frequency domain, `plot [:1e10][1e-8:] 'output' u 1:($2/2.41726e-05) w l lw 7` in `gnuplot` will produce Fig. 8.2. The time domain signal after the 1500 timesteps is not zero in amplitude. When the presentation in `gnuplot` avoids the time domain data upto 3000 time steps, `plot [3000:] 'timedata' w l lw 7` will present Fig. 8.3 which shows a clear resonance signal. The signal in Fig. 8.3, *i.e.* the time domain signal last 2000 time steps is turned into the frequency domain and saved as `output`. `plot [:1e10][1e-8:] 'output' u 1:($2/1.07503e-09) w l lw 7` in `gnuplot` will produce Fig. 8.4. Fig. 8.4 clearly shows the resonance frequency.

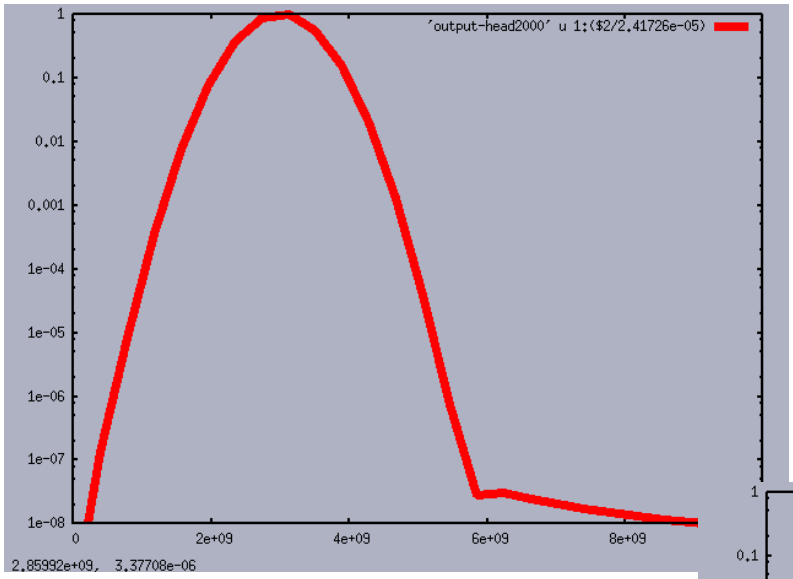


Figure 8.2: Frequency spectrum of Fig. 8.1

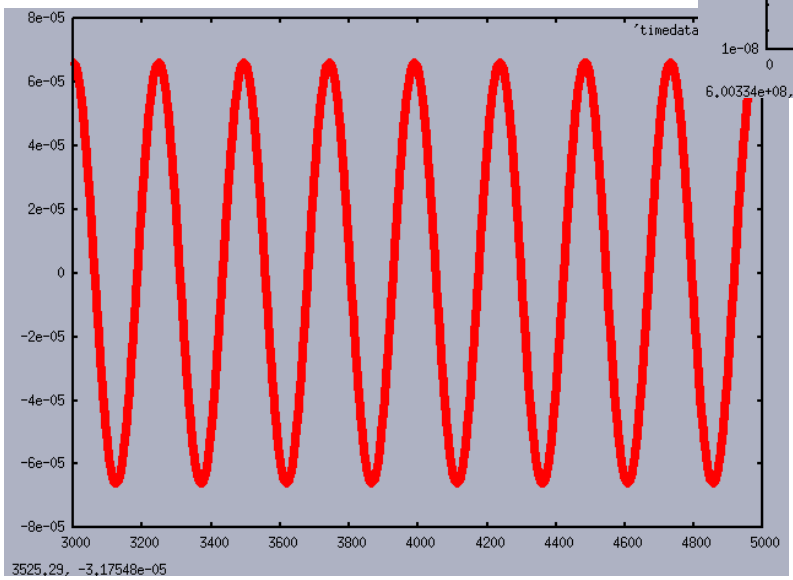


Figure 8.3: Time domain signal only after 3000 time steps

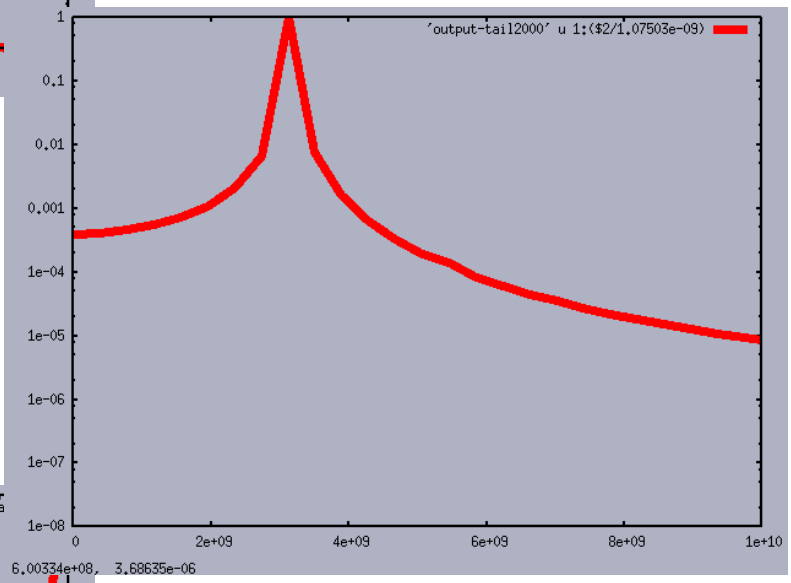


Figure 8.4: Frequency spectrum of Fig. 8.3

8.2 Assignment on shell script with awk and shift

Run a command `script` at the beginning of your work under Linux. This log of your usage of unix environment is the part of the assessment.

`PECnew.tar.gz` has `E_field_0????.out` which have 4 columns. First, second and third columns are x, y and z values, respectively and fourth column has $\mathbf{E}$ field value. This data is the result of the FDTD calculation with $\mathcal{X} = 285$, $f =$ 3GHz. The dimensions of the FDTD space are $190 \times 190 \times 190$. The FDTD space is excited at ($95\Delta x$, $95\Delta y$, $95\Delta z$). Time domain signal needs to be extracted.

- Create a directory called `assignment8` under your home directory and go to the directory.
- Untar the datafile `PECnew.tar.gz` into the current directory by `tar -zxvf /coursedisk/course/skeleton/PECnew.tar.gz`
- Write a shell script named `plotshell` which
  - takes arguments including x-value, y-value and z-value, and column number which needs to be extracted, the name of data files which could be expressed as `E*out`
  - produces the time domain signal and saves the ascii datafile as `plot.data` . The format of `plot.data` should be the following:
    - there are two columns
    - the first column has time steps (Hint: `ARGIND` in `awk`)
    - the second column has $\mathbf{E}$ values
- Run your shell script to extract a time domain signal at (a+30,95,95) where a is the username number. For example, if your username is eee15, you should obtain the time domain signal at (45,95,95) from `PECnew.tar.gz`.
- Plot the time domain signal after 4000 time steps with x-axis time in second and y-axis voltage using `gnuplot` and turn it into the pdf file through `tgif` treatment. Name the resultant pdf file `timedomainsignal.pdf`.
- type `exit` at the end of your usage of Linux. This will produce a file called typescript at the current directory. Please change the name of the typescript to, for example, typescript-chap8 which should be in `/YourHomeDirectory/assignment8/`, not any other subdirectories such as `/YourHomeDirectory/assignment8/subdirectory`.

Assignment should present the shell script `plotshell`, `typescript` which shows the execution of the shell script and `timedomainsignal.pdf`.

```
#!/bin/bash
if [ $# -lt 5 ]
then {
   echo "Prints a point for all the time steps"
   echo "Parameters: x y z fieldNumber files"
   echo " "
   exit 0
   }
fi
x=$1
y=$2
z=$3
fieldNumber=$4
shift 4
gawk -v x=$x -v y=$y -v z=$z -v field=$fieldNumber '($1==x && $2==y && $3==z){print ARGIND" "$field}'
 $@ > plot.data
plotPoint.sh lines 1-15/15 (END)
```

Figure 8.5: Example answer of plotshell

– Location (x,y,z coordinate) is stored in variables

[2 marks]

– Proper usage of ARGIND or other method to produce the first column

[1 mark]

– Proper production of the second column

[1 mark]

The shell script plotshell should include most of Fig. 8.5 The log.pdf should show the execution of the shell script which takes arguments

[2 marks]

The time domain signal in timedomainsignal.pdf should show the resonance signal exclusively.

[2 marks]

Since each student records the time domain signal at the different location, the highest value of the resonant signal changes depending on the student. Table 8.1shows this highest value of the resonant signal depending on the each username number.
The highest value of timedomainsignal.pdf which each student produces should match to Table 8.1

[2 marks]

username number	maximum value of the resonant signal in timedomainsignal.pdf
1	2.6582936E-06
2	2.8209340E-06
3	2.9824585E-06
4	3.1423531E-06
5	3.3010476E-06
6	3.4603424E-06
7	3.6176450E-06
8	3.7746129E-06
9	3.9304732E-06
10	4.0858163E-06
11	4.2375409E-06
12	4.3897057E-06
13	4.5416682E-06
14	4.6921145E-06
15	4.8403535E-06
16	4.9873852E-06
17	5.1338802E-06

username number	
18	5.2780774E-06
19	5.4221578E-06
20	5.5629803E-06
21	5.7029847E-06
22	5.8419423E-06
23	5.9792119E-06
24	6.1153428E-06
25	6.2490981E-06
26	6.3807906E-06
27	6.5107547E-06
28	6.6394882E-06
29	6.7659194E-06
30	6.8911281E-06
31	7.0141518E-06
32	7.1349045E-06
33	7.2543271E-06
34	7.3722590E-06
35	7.4869045E-06
36	7.6003930E-06

username number	
37	7.7115519E-06
38	7.8210232E-06
39	7.9288702E-06
40	8.0326663E-06
41	8.1356329E-06
42	8.2365623E-06
43	8.3342738E-06
44	8.4296980E-06
45	8.5240154E-06
46	8.6147020E-06
47	8.7034423E-06
48	8.7913977E-06
49	8.8760144E-06
50	8.9572059E-06
51	9.0369958E-06
52	9.1150341E-06
53	9.1881793E-06
54	9.2610517E-06
55	9.3301242E-06
56	9.3967637E-06

Submission of the assignment for EEEN40063

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Student signature	

.....

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Recipient signature	

Chapter 9

Numerical dispersion

9.1 Waveform change observation

200 ×200 ×200 dimensional FDTD space is excited at the centre (100,100,100). FDTD space is filled with a lossy medium (σ = 0.7 S/m, ϵ_∞ = 4, ϵ_S = 6, $\tau_D = 7{\cdot}10^{-11}$). The waveform of the source excitation is the Gaussian pulse in Eq. 9.1.

$$GS(t) = \exp\left[-\left(\frac{8wf_{CW}t}{\Delta t}\right)^2\right]\Pi(wf_{CW}t)\cos(2\pi\alpha f_{CW}t) \quad (9.1)$$
$$= \exp[-(8\sqrt{3}wf_{CW}^2\mathcal{X}t)^2]\Pi(wf_{CW}t)\cos(2\pi\alpha f_{CW}t)$$

where $\Pi(t) = \begin{cases} 1: & |t| < \frac{1}{2} \\ 0: & |t| > \frac{1}{2} \end{cases}$ and w is used to change the pulse width. This pulse duration is

$$\frac{-1}{2wf_{CW}} \leq t \leq \frac{1}{2wf_{CW}} \quad (9.2)$$

$\mathcal{X} = 150$ and $f_{CW} = 3$GHz and $w = 2$ and $\alpha = 6$.

`waveform.tar.gz` has `fdtdobservation-???.data` and `theoreticalobservationwaveform-???.data` where `???` is from 101 - 150.

`fdtdobservation-???.data` has the normalised time domain signal received at (???, 100,100) and `theoreticalobservationwaveform-???.data` is the normalised time domain waveform at (???,100,100) if the signal is received with no numerical dispersion.

When `fdtdobservation` and `theoreticalobservationwaveform` are plotted in the same figure, the waveform difference can be observed clearly.

Plotting both files can be achieved by the following shell script.

```
 1  #!/bin/csh
 2  rm gnufile
 3  touch gnufile
 4  set k = 101
 5  while($k  < 151)
 6  set filename = fdtdobservation-$k
 7  echo "plot [:5e-10] '$filename'  w l lw 3 "  >> gnufile
 8  set filename = theoreticalobservationwaveform-$k
 9  echo "replot  '$filename'  w l  lw 2"  >> gnufile
10  echo "pause 1" >> gnufile
11  @ k ++
12  end
13  echo "pause -1 " >> gnufile
14  gnuplot  gnufile
15  exit 0
```

The major steps of the script include

line 3 creates a file `gnufile`

lines 4 -13 write out the gnuplot commands into `gnufile`

line 14 perform gnuplot with the command in `gnufile`

By producing a file which is a collection of gnuplot commands, figure production is automated.

9.2 Assignment on shell script with DFT

PECnew.tar.gz contains E_field_0????.out which have 4 columns. First, second and third columns are x, y and z values, respectively and fourth column has **E** field value. This data is the result of the FDTD calculation with $\mathcal{X} = 285$, $f =$ 3GHz. The dimension of the FDTD space is 190×190×190. This FDTD space is terminated by metal wall to simulate a cavity. The FDTD space is excited at ($95\Delta x$, $95\Delta y$, $95\Delta z$). Theoretical resonant frequency can be obtained analytically. Resonant frequency should be measured using the time domain signal extracted.

- Run script to record all of your activities.
- Create a directory called assignment9 under your home directory and move to the directory.
- Untar the datafile PECnew.tar.gz into the current directory by
  tar -zxvf /coursedisk/course/skeleton/PECnew.tar.gz
- Write a shell script named plotshell which
  - takes arguments including x-, y- and z-values, target column number which need to be extracted, and names of data files, which could be expressed as E*out
  - produces and saves ascii datafile out of E_field_0????.out as a file called plot.data .
- Produce a file initialdata whose format is

```
multiplifier, 1
element_per_wavelength,    285
wave_frequency,  3e9
maxsteps,    5000
```

  maxsteps should be modified depending on the number of data used.

- Produce a file timedata from plot.data which is the time domain signal record at (a,50,50) where a is the username number. timedata should have only second column of plot.data and ideally should have only resonant signal without the direct signal from the source. Use tail to extract the latter part of the signal. For example if the last 123 timestep data need to be extracted, run
  tail -123 plot.data |
  awk '{print $2}' > timedata.
- Run ./calspectrum to produce real and imaginary parts of the spectrum as outputreal and outputimag.
- paste outputreal outputimag|
  awk '{print $1, $2**2+$4**2}' > output
  will produce the spectrum magnitude.
- Plot obtained spectrum with gnuplot .
- Change the length of the timedata three times and save the frequency spectrum from each time domain signal such as
  output123, output1123 and output2123 where 123, 1123, 2123 are the length of the time domain signal. The data length should be changed three times based on Table 9.1.

Table 9.1: Data length of the time domain signal

User number	length of time domain signal
1	210 ,1210 ,2210
2	320 ,1320 ,2320
3	401 ,1401 ,2401
4	123 ,1123 ,2123
5	420 ,1420 ,2420
6	134 ,1134 ,2134
7	350 ,1350 ,2350
8	306 ,1306 ,2306
9	432 ,1432 ,2432
10	504 ,1504 ,2504
11	531 ,1531 ,2531
12	603 ,1603 ,2603
13	801 ,1801 ,2801
14	127 ,1127 ,2127
15	235 ,1235 ,2235
16	146 ,1146 ,2146
17	380 ,1380 ,2380
18	623 ,1623 ,2623
19	704 ,1704 ,2704
20	137 ,1137 ,2137
21	372 ,1372 ,2372
22	471 ,1471 ,2471
23	516 ,1516 ,2516
24	543 ,1543 ,2543
25	157 ,1157 ,2157
26	247 ,1247 ,2247
27	256 ,1256 ,2256
28	346 ,1346 ,2346
29	149 ,1149 ,2149
30	239 ,1239 ,2239
31	653 ,1653 ,2653
32	815 ,1815 ,2815
33	167 ,1167 ,2167
34	186 ,1186 ,2186
35	483 ,1483 ,2483
36	726 ,1726 ,2726
37	753 ,1753 ,2753
38	852 ,1852 ,2852
39	178 ,1178 ,2178
40	259 ,1259 ,2259
41	268 ,1268 ,2268
42	358 ,1358 ,2358
43	367 ,1367 ,2367
44	457 ,1457 ,2457
45	467 ,1467 ,2467
46	683 ,1683 ,2683
47	278 ,1278 ,2278
48	639 ,1639 ,2639
49	765 ,1765 ,2765
50	837 ,1837 ,2837
51	739 ,1739 ,2739
52	578 ,1578 ,2578
53	569 ,1569 ,2569
54	678 ,1678 ,2678
55	869 ,1869 ,2869
56	879 ,1879 ,2879

- Plot these (such as `output123`,`output1123`, `output2123` for the user eee1) in a gnuplot window within the frequency range of $0 \sim 10$ GHz and decorate in tgif. The maximum value of each spectrum should be 1. Convert the tgif file to the pdf file and name the pdffile `freqspacing.pdf`.

- Discuss the observation in a file called `freqspacing`.

Assignment report should have `freqspacing.pdf` and `freqspacing`. Do not delete the files you produced such as
`output123`,`output1123`, `output2123` as they are assessed electronically.

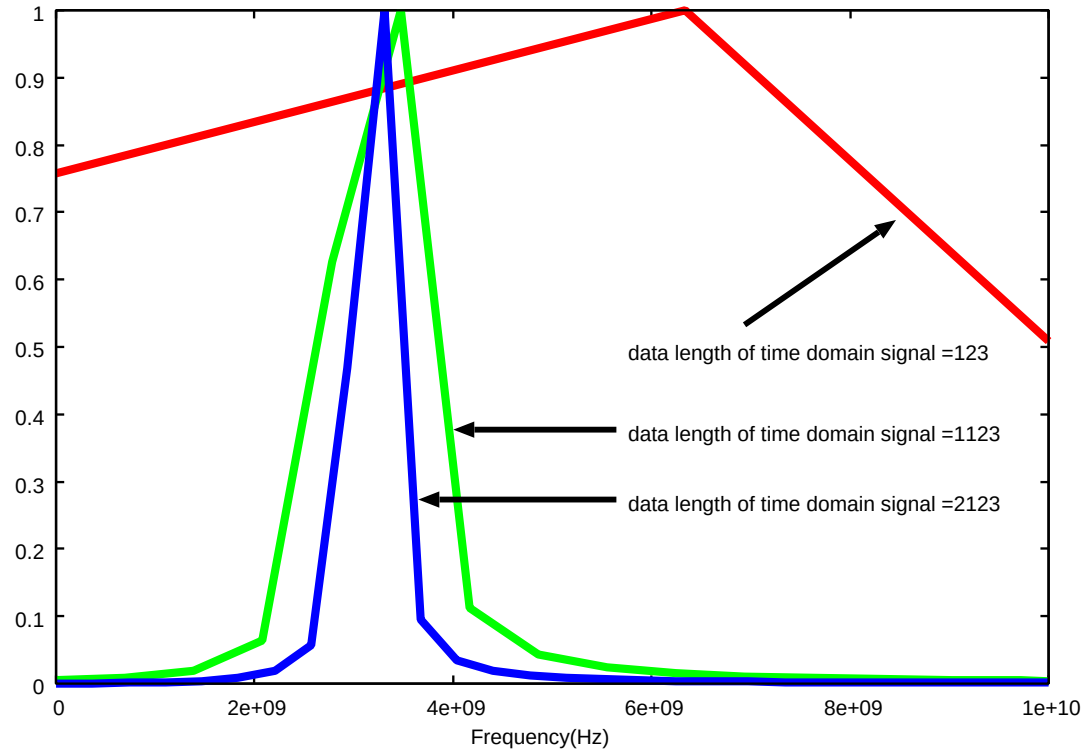


Figure 9.1: `plot[:1e10]'output123'u1:($2/6.24107e-10)wl, 'output1123'u1:($2/5.49297e-10)wl, 'output2123'u1:($2/6.15473e-10)wl` for the signal obtained at (25,50,50).

The length of three files output??? each user produces should be identical to the length presented in Table 9.1.

[2 marks]

Plotshell is in Fig. 8.5. One of the examples of `freqspacing.pdf` is Fig. 9.1. freqspacing.pdf should have three lines

[1 mark]

which have correct data lengths based on Table 9.1 and the spectrum which is obtained from time domain signal with long data length has fine Δf

[2 marks]

and the maximum value of each spectrum is one

[1 mark]

Discussion should include the following:

- Theoretical resonant frequency is about 3.18GHz

  [2 marks]

- Depending on the frequency discretisation, the difference between the resonant frequency observed and the theoretical resonant frequency changes

  [2 marks]

Submission of the assignment for EEEN40063

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Student signature	

.....

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Recipient signature	

Chapter 10

Wave propagation

10.1 Movie production

The visualization of the result helps the software debugging as well as the understanding of physical phenomena through numerical simulation. In Section 7.2, ppm files were successively displayed via `xv`. There is an alternative simple way to produce a movie. Some unix commands such as `convert` and `ppmtompeg` convert multiple ppm files into a single mpeg file. Here, a special care must be taken for the name of the files. For example, in a directory which has 11 files of test1.ppm, test2.ppm, test3.ppm, test4.ppm, test5.ppm, test6.ppm, test7.ppm, test8.ppm, test9.ppm, test10.ppm and test11.ppm,

```
convert -delay 1 test*.ppm out.mpg
```

will produce a mpeg file which displays these ppm files in the wrong order of test1.ppm, test10.ppm and test11.ppm test2.ppm,test3.ppm,test4.ppm, test5.ppm,test6.ppm,test7.ppm, test8.ppm,test9.ppm. When files test10.ppm and test9.ppm are compared in listing, both of files have identical four letters and fifth character differs between these two files. Number 1 comes earlier than 2∼9 in listing. This is the reason why the file display order was different from the order of the time steps.

A command `ppmtompeg parameterfile` will not face the problem mentioned above as long as `parameterfile` specifies `test*.ppm [1-11]` as the input files. An example of the parameterfile is

```
PATTERN         IBBPBBPBBPBBPBBP
OUTPUT          output.mpg
BASE_FILE_FORMAT        PPM
YUV_SIZE        352x240
INPUT_FORMAT UCB
INPUT_CONVERT   *
GOP_SIZE        16
SLICES_PER_FRAME        1
INPUT_DIR       /home/fumie/graphics/testdata
INPUT
test*.ppm       [1-804]
END_INPUT
PIXEL           FULL
RANGE           10
PSEARCH_ALG     LOGARITHMIC
BSEARCH_ALG     CROSS2
IQSCALE         8
PQSCALE         10
BQSCALE         25
REFERENCE_FRAME ORIGINAL
BIT_RATE  1000000
BUFFER_SIZE 327680
FRAME_RATE 23.976
```

During conversion of the file formats, `convert` requires a vast working space. In tcsh,

```
setenv MAGICK_TMPDIR  /some/large/diskspace/director
```

, in bash,

```
export MAGIC_TMPDIR=/some/large/diskspace/directory
```

specifies the working diskspace for `convert`.

`ppmtopmeg` does not need significant amount of memory nor time to produce a mpeg file as long as the `parameterfile` is properly written.

Table 10.1: Open Data file and view plane allocation

user number	file name	plane to see
1	data-10-1-1-0-open.tar.gz	z
2	data-0-20-20-0-open.tar.gz	x
3	data-1-1-1-0-z-open.tar.gz	z
4	data-0-20-20-0-x-open.tar.gz	x
5	data-1-1-1-0-x-open.tar.gz	z
6	data-0-20-20-0-z-open.tar.gz	x
7	data-1-1-1-0-open.tar.gz	z
8	data-0-30-30-0-open.tar.gz	x
9	data-0.1-1-1-0-z-open.tar.gz	z
10	data-0-30-30-0-x-open.tar.gz	x
11	data-0.1-1-1-0-x-open.tar.gz	z
12	data-0-30-30-0-z-open.tar.gz	x
13	data-0.1-1-1-0-open.tar.gz	z
14	data-0-5-5-0-open.tar.gz	x
15	data-0-5-5-0-z-open.tar.gz	z
16	data-0-5-5-0-x-open.tar.gz	x
17	data-0-5-5-0-x-open.tar.gz	z
18	data-0-5-5-0-z-open.tar.gz	x
19	data-0-5-5-0-open.tar.gz	z
20	data-0.1-1-1-0-open.tar.gz	x
21	data-0-30-30-0-z-open.tar.gz	z
22	data-0.1-1-1-0-x-open.tar.gz	x
23	data-0-30-30-0-x-open.tar.gz	z
24	data-0.1-1-1-0-z-open.tar.gz	x
25	data-0-30-30-0-open.tar.gz	z
26	data-1-1-1-0-open.tar.gz	x
27	data-0-20-20-0-z-open.tar.gz	z
28	data-1-1-1-0-x-open.tar.gz	x
29	data-0-20-20-0-x-open.tar.gz	z
30	data-1-1-1-0-z-open.tar.gz	x

10.2 Assignment on shell script with mpeg production

The tar file `data-a-b-b-0-(x,z-)open.tar.gz` has `E_field_0????.out` which have 4 columns as is the case with Section 6.2. First, second and third columns are x, y and z values, respectively and fourth column has $\mathbf{E}$ field value. This data is the result of the FDTD calculation with $f = 3$GHz. The dimension of the FDTD space is $150 \times 150 \times 150$. The FDTD space is excited at the centre of FDTD space. The field movement should be observed by creating a mpeg file or a shell script to view the ppm in a continuous manner.

- Run `script` before starting your work.
- Create a directory called `assignment10` under your home directory and move to the directory.
- Untar the datafiles `data-a-b-b-0-(x,z-)open.tar.gz` into the current directory. Table 10.1 shows the name of the data file each user is supposed to use and the view plane.

user number	file name	plane to see
31	data-0-20-20-0-open.tar.gz	z
32	data-10-1-1-0-open.tar.gz	x
33	data-0-2-2-0-z-open.tar.gz	z
34	data-10-1-1-0-x-open.tar.gz	x
35	data-0-2-2-0-x-open.tar.gz	z
36	data-10-1-1-0-z-open.tar.gz	x
37	data-0-2-2-0-open.tar.gz	z
38	data-100-1-1-0-open.tar.gz	x
39	data-0-10-10-0-z-open.tar.gz	z
40	data-100-1-1-0-x-open.tar.gz	x
41	data-0-10-10-0-x-open.tar.gz	z
42	data-100-1-1-0-z-open.tar.gz	x
43	data-0-10-10-0-open.tar.gz	z
44	data-1000-1-1-0-open.tar.gz	x
45	data-0-1-1-0-open.tar.gz	z
46	data-1000-1-1-0-x-open.tar.gz	x
47	data-1000-1-1-0-x-open.tar.gz	z
48	data-0-1-1-0-open.tar.gz	x
49	data-1000-1-1-0-open.tar.gz	z
50	data-0-10-10-0-open.tar.gz	x
51	data-100-1-1-0-z-open.tar.gz	z
52	data-0-10-10-0-x-open.tar.gz	x
53	data-100-1-1-0-x-open.tar.gz	z
54	data-0-10-10-0-z-open.tar.gz	x
55	data-100-1-1-0-open.tar.gz	z
56	data-0-2-2-0-open.tar.gz	x
57	data-10-1-1-0-z-open.tar.gz	z
58	data-0-2-2-0-x-open.tar.gz	x
59	data-10-1-1-0-x-open.tar.gz	z
60	data-0-2-2-0-z-open.tar.gz	x

For example, if your username is `eee15`, use `data-0-5-5-0-z-open.tar.gz` to visualise $z = 75$ plane. It is important that you take the right dataset so that proper assessment can be performed. Each dataset has different information.

- Write a shell script named `mpgshell` which
  - produces a ppm file from the files `E_field_0????.out`
  - performs necessary environment setting
  - converts the ppm files into mpg file called `out.mpg`
- Send the mpg file you produced to your email account
- Receive the mpg file in the Window's web browser
- View the mpg file you created to see if `mpgshell` performs what you wanted the shell script to do

The assignment report should consist of `mpgshell`. Keep all the ppm files and mpeg file you have created. They are going to be assessed electronically You do not have to use `convert` or `ppmtompeg` when you find better solutions.

Each student's answer should be identical to the answers in `130.88.154.35:/local/fumietest-*-movie`.

[2 marks]

If convert is used the shell script should include Fig. 7.1 and Fig. 10.1. If ppmtompeg is used, the parameterfile should be written in a correct and efficient manner. The model answer *i.e.,* the ppm files and out.mpg which each user should produce are `130.88.154.35:/local/fumietest-*-movie`.

- Correct order of the ppm files presented

[2 marks]

- Correct number of ppm files presented

[2 marks]

are the checking points of out.mpg

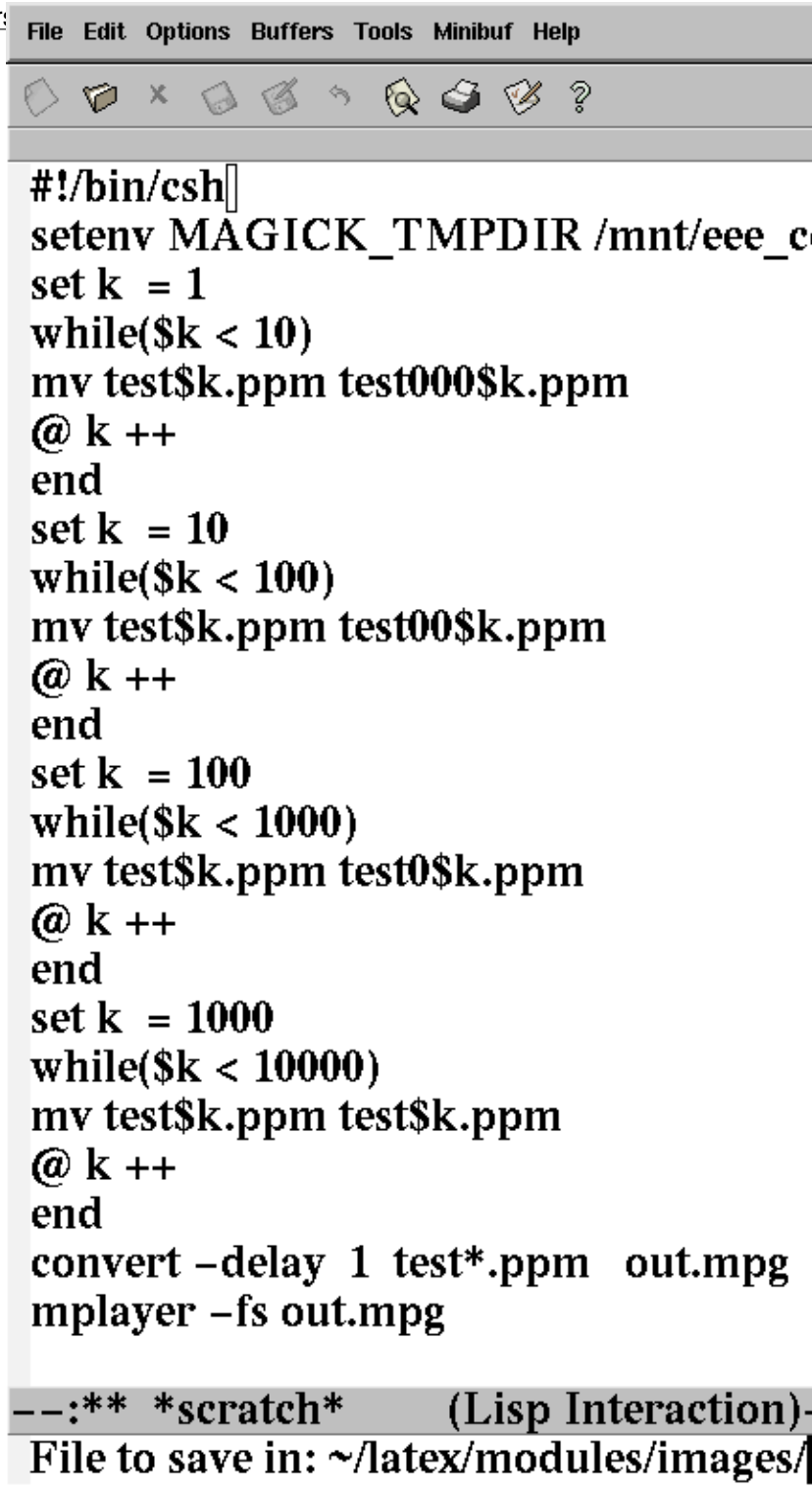


Figure 10.1: Example of mpgshell
Checking points:

- Setting of the MAGICTMPDIR

[2 marks]

- Proper usage of convert

[2 marks]

Submission of the assignment for EEEN40063

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Student signature	

.....

Assignment number	
Username	
Student name	
Student ID number	
Date and time	
Recipient signature	