Polynomials and Rational Functions:

```
(C1) P(x) = 260*x^6 - 3725*x^5 + 1135*x^4 +
            520*x^3 - 7450*x^2

(C2) factor (P(x));

(C3) (4*x - 17) * (x+1), expand;
```

Solving polynomials:

```
(C4) a*x^2 + b*x + c = 0;

(C5) solve (%, x);
```

(nifty isnt it ?!)

Expansions:

```
(C6) taylor ( sin(x), x, 0, 5 );
```

Taylor series expansion of sin(x) around point 0 upto power 5.

Linear Algebra:

```
(C1) m : matrix ([1, 2, 3], [4, 5, 6], [7, 8, 9]);
```

```
(C2) determinant (m);
```

(its singular :))

```
(C3) m^^-1;
```

To see results in floating point:

```
(C4) numer : true;
```

```
(C5) charpoly (m, x);
```

```
(C6) solve (%, x);
```

Try some symbollic products too !

Limits:

The following predefined sumbols are necessary:

INF: positive infinity
MINF: negative infinity
UND: undefined
IND: indefinite, not bounded
INFINITY: complex infinity

```
(C1) limit (1 / (x - 2), x, 2, minus);

(C2) limit (sin(x), x, 0);

(C3) limit (cos(x), x, INF);
(D3)                                        ind
```

Integration:

```
(C1) integrate (cos(3*x), x);
```

if Maxima cant integrate, it will simply return the function under an integral sign. Also, depends wont work for integration.

The general form of integrate is:

integrate (f1(x,y),	x,	0,	1);
	w. r. t	from	to

Functional dependencies can be represented as follows:

Say

$f = f(x, y)$
$g = g\ (x, y)$
$r = r(u, v, w)$
$s = s(u, v, w)$
$u = u(t)$

To declare these dependencies:

```
(C11) depends([f, g], [x, y], [r, s], [u, v, w], u, t);
```

why do we need this?

Try the following:

```
(C12) diff (r, t);
```

make sense now?

Defining a function:

```
(C1) f(x) := x^2;
```

```
(C2) f(2);
```

Differentiation:

```
(C3) diff (sin(x),x);
```

if you want to differentiate it 4 times say, try:

```
(C4) diff (sin(x),x, 4);
```

if you want to differentiate it with respect to different variables:

```
(C5) diff (sin(x*y), x, 2, y, 1);
```

The % sign can be used to refer to the last displayed item.

Using MAXIMA.

Maxima can do a lot of very useful things:

- ❏ Limits
- ❏ Differentiation
- ❏ Integration (definite and indefinite)
- ❏ ODEs
- ❏ Factorization
- ❏ Taylor Series
- ❏ Laplace Transforms
- ❏ Linear Algebra
- ❏ FORTRAN Generation.

Plotting three-dimensional plots with surfaces or contours

By default, a splot command plots only the surface and not a contour of a surface. However, by using the set contour command, you can have gnuplot plot a contour of the function.

Try the following:

```
splot sinc(x) * sinc(y)
set contour base; splot sinc(x) * sinc(y)
set contour surface; splot sinc(x) * sinc(y)
set contour both; splot sinc(x) * sinc(y)
```

Plotting three-dimensional functions

The syntax of the splot command for a three-dimensional function is as follows:

```
splot {ranges} function {title “title text”} {style}

splot sin(x) * cos(y)
```

Use

```
set hidden
```

to hide invisible surfaces.

To control number of isolines (wire frame), use

```
set isosamples
```

Plotting three-dimensional data files

The syntax of the splot command for plotting a three-dimensional set of data is as follows:

```
splot {ranges} “datafile” {using data-format}
      {index i} {title “title text”} {style}
```

In the case where all three coordinates are specified, you must use the set parametric command to set gnuplot to use the plot parameters. For both cases, a blank line is used to separate different sets of data.

```
set parametric
splot “glass.dat” with lines
```

Other available plot modes

❑ Parametric Plots

Instead of using a single function to describe a curve, you can use the set parametric command to set Gnuplot to use parametric curves. When making these plots, you will need to specify two functions of the dummy variable t, called f(t) and g(t). (To turn it off, use set noparametric)

```
set parametric
plot cos(t * 3), sin(t * 2)
```

❑ Polar Plots

Similar to the above setting, you can use the set polar option to set the axes to use polar coordinates, rather than cartesian. For polar plots, the x coordinate becomes theta and the y coordinate becomes r or radius. (To turn it back to cartesian coordinates, use set nopolar)

```
set polar
plot cos(2 * x)
```

Function plots

The syntax of the plot command for a two-dimensional function is as follows:

```
plot {ranges} function {title “title text”}
     {style}
```

eg.

```
sinc(x) = sin(pi * x) / (pi * x)
set samples 25; plot sinc(x)
set samples 500; plot sinc(x)
```

Samples allows you to set the sampling points (default is 100 points)

Two-dimensional plots: Data plots/Error bar plots

The syntax of the plot command for plotting a two-dimensional set of data is as follows:

```
plot {ranges} “datafile” {using data-format}
     {title “title text”} {style}
```

If several plots are to be plotted, subsequent data plots (or functions) can be specified, separated by commas.

Error bar plots

For plots with error bars, the data file used also needs the information for the error bars. If only a third column is given, that column is used for the delta values of Y. If the third and fourth columns are specified, the columns are used for the low value for Y and high value for Y, resp.

```
plot “fileerr.txt” with lines, “fileerr.txt”
with errorbars
```

Using Gnuplot

Gnuplot allows both data and function plots, and also allows many plots to be overlaid. In addition, you can define your own functions and use data files.

Plot types available

- ❏ Two-dimensional plots
  - Data plots/Error bar plots
  - Function plots
  - Other available plot modes
- ❏ Three-dimensional plots
  - Data plots
  - Function plots
  - Surface/Contour

Using xfig:

Xfig is a very useful tool for draqing pictures for your papers.

(Run Xfig and familiarize yourself with it thoroughly).

To incorporate xfig figures into latex, export the xfig figures as encapsulated postscript (eps, portrait). Say you exported it into a file figure.eps. To get this into a latex document do the following:

```
\begin{figure}[htb]
\centerline{
\epsfysize=4.5in %the x-size of the figure
\epsfxsize=3.0in %the y-size of the figure
\epsfbox{figure.eps}
}
\caption{This is a sample figure.}
\label{sample_figure}
\end{figure}
```

To refer to the figure, simply say Figure \ref{sample_figure} and latex will take care of crossreferencing figure numbers etc.

Font Typeface Changes

Changing the font from regular characters, to italics, or bold or slant, or even sans serif. LaTeX has 6 typefaces as standard to all fonts. Those typefaces are : (typeface.tex)

\rm - roman (regular characters)
\sf - sans serif
\sl - slanted roman
\it - italics
\sc - small caps
\tt - typewriter

Look at special.tex for description of special characters and sequences.

Font Size Changes

The relative font size can be changed with a simple declaration. The normal font size is, not too surprisingly, \normalsize. In increasing order of size: (size.tex)

\tiny
\scriptsize
\footnotesize
\small
\normalsize
\large
\Large
\LARGE
\huge
\Huge

You will notice that in the example , the font changes are made within the {} braces. The braces in this case act as scoping, delimiting the range of the effect of the font change.

❑ Text Formatting Environments

center - defines area that will be centered (center.tex)
flushleft - defines area that will be left justified (flushleft.tex)
flushright - defines area that will be right justified (flushright.tex)
quote - defines area that will be block indented, paragraphs separated (quote.tex)
quotation - defines area that will be block indented (quotation.tex)
verse - defines area for poetry-like style (verse.tex)

❑ List Making Environments

itemize - defines area for bulleted list (itemize.tex)
enumerate - defines area for numbered list (enumerate.tex)
description - defines area for definition-type list (description.tex)

❑ Table Making Environments

tabbing - defines area for tabbing (tabbing.tex)
tabular - defines area that will make a structured table (tabular.tex)

❑ Mathematical Typesetting Environments

array - environment for typesetting arrays and matrices
math - environment for typesetting mathematical equations
displaymath - environment for typesetting mathematical equations
equation - environment for typesetting mathematical equations

Section Declarations

Section declarations are used by LaTeX to split up a document into logical sections, and to give points in the document that LaTeX can add to a table of contents. Sections can be a variety of types:

- \part
- \chapter
- \section
- \subsection
- \subsubsection
- \paragraph
- \subparagraph

(download, compile and look up second.tex, third.tex).

LaTeX comes with several standard document styles:

- article - for those wanting to publish articles in journals
- letter - for those letters to potential employers
- report - for those science reports you hate rewriting
- book - for those who are really ambitious
- slides - for your T.A. to make really impressive overheads

As well, many other styles have been written by LaTeX enthusiasts:

- thesis - the theses style
- resume - a couple of styles which help create resumes
- music - styles to allow you to typeset music.

Many more are available.

Text formatting using latex:

To convert the dvi file to ps, type:

```
dvips -o first.ps first
```

or sometimes

```
dvi2ps -o first.ps first
```

Here first.ps is the desired postscript file that can be printed on the printer using

```
lpr -P<printername> first.ps
```

Text formatting using LaTex:

Copy the file first.tex from
http://www.cs.purdue.edu/homes/ayg/CS590C/latex/first.html

The file looks as follows:

```
\documentstyle{article}
\begin{document}
This is my very first document.$^{not}_{really!}$
\end{document}
```

To compile the text file, type

```
latex first
```

(the .tex extension is the default)
This creates several files: text.dvi (a device independent output file, a .err file (errors), .log (log), .aux and .bbl (bibliography).

CS590C: Introduction to Computational Science

Lecture 4: Tools for symbolic math and presentations.

This lecture is intended to familiarize you with:

- Test processing tools: LaTex, Framemaker.
- Graphics tools: gnuplot, xfig
- Tools for symbolic math: MAXIMA.