III. DOCUMENT PREPARATION USING TEX

One way of avoiding problems with examinations and other documents is to prepare them in a careful and reliable manner. The mathematical typesetting program **TEX** is an extremely powerful tool for creating scientific text complete with formatting and symbols. A thorough discussion of **TEX** is beyond the scope of this course. Fortunately, one can do a great deal with very little background, so the aim here is to provide the minimum amount that is needed for standard tasks that a teaching assistant might want to complete in connection with his or her teaching assignments.

There are numerous versions of **TEX** with a wide variety of extra packages to take care of special needs. We shall deal exclusively with the so-called **plain TEX** program, which is more than adequate for the sorts of documents we shall discuss.

There are numerous books and web sites that are devoted to **TEX.** The online directory for this course (**`~chair/math302`**) contains two relatively concise introductions to **TEX** in both **`ps`** and **`pdf`** formats:

A Gentle Introduction to **TEX** , by Michael Doob
(**`~chair/math302/gentle-tex.ps`** and **`~chair/math302/gentle-tex.pdf`**)

TEX *made easy*, by Daniel M. Zirin
(**`~chair/math302/tex-made-easy.ps`** and
`~chair/math302/tex-made-easy.pdf`)

Both of these were downloaded from the web. They are ten years old and a bit dated, but they contain everything that is needed to start and do not get into other issues that can be distracting to a beginner.

Many of us learned to use **TEX** by starting out with an existing document as “boiler plate” and modifying it so that it served our purposes. This is the approach that will be taken here. Four old documents of mine will be used as examples. Here are brief descriptions of them and indications of their significance.

A. *A handout with course guidelines that one might distribute at the beginning of a course.* — This is an example of a document that is mostly text with only a few mathematical pieces.
B. *An outline of the material to be covered in the course along with homework assignments*. — This is another example of the same kind.
C. *A sample examination for the course.* — In addition to providing an example where more mathematical expressions are involved this also shows how one performs various tricks needed to produce an exam that will be easy to work with for both the students and those who grade the papers; it also provides examples of how to set up matrices.
D. *A handout discussing some supplementary material for another course.* — This provides some different examples of mathematical expressions taken from calculus rather than linear algebra.

Both the original files and the printed output from such files are contained in the course directory; there will be comments on some point in each file that seem particularly noteworthy. However, before doing this it is appropriate to say a few words about the creation and processing of **TEX** files.

Creating a document using TEX on the Department's computer network

The first step is to create an ordinary ASCII or text file with suitable input. For example, the following **TEX** file can be used to create a document that consists only of the word "Hello" on the top line:

```
Hello
\end
```

The first line just gives the input exactly as it should appear on the document, and the second line is a control command that terminates the typesetting program; all control commands in **TEX** are preceded by backslashes as in this line. The name for the file must end with the extension `.tex`, so `hello.tex` would be a perfectly good name for this file.

The next step is to convert this human readable file into a machine readable file called a `.dvi` file. To do this, enter the command

```
tex hello
```

which will create two files, one of which is the desired file `hello.dvi` and the other of which is a text file `hello.log` that records some basic information about the creation of the `.dvi` file. If you are working directly on a computer that has some sort of windows system (in other words, you are not on a telnet connection), then you can preview the file on the video display using the command `xdvi hello.`

Of course, mistakes are going to happen when you create files and try to process them. In particular, the `.log` file will list various sorts of problems or mistakes, including mistakes that bring the text processing program to a halt. In these cases one has to go back to editing the original document and trying again. Other mistakes are likely to be noticed when you preview the document, and it is also likely that you will see things that do not look as good as you would like. Correcting both types of problems will require going back to the original text file.

At this point you can print out the document with the following command

`print -P`[*printername*] **`hello.dvi`**

and pick up a hard copy of the document from the printer output. Of course, in many cases you might want to have online copies of the documents instead of — or in addition to — the hard copy. The `dvi` file is an example of this sort, but one disadvantage is that many computers do not have the software needed to display such documents. Therefore in many cases you might prefer to create either a PostScript file `hello.ps` or an Acrobat Portable Document File `hello.pdf` in addition to the files that already exist; the latter is especially appealing because free copies of the Acrobat Reader are easy to download and very widely used on all sorts of computer systems.

A PostScript file can be created using the following command:

```
dvips -f  hello > hello.ps
```

One can then create the file **`hello.pdf`** by entering the command **`ps2pdf hello.ps`** and afterwards it is probably a good idea to check that everything worked by viewing the newly created **`pdf`** file.

Comments on the sample documents

DOCUMENT A. The default type size for **TEX** is 10 points (1 inch = 72 points). For many purposes this is too small, but for the current document it is just the right size so that everything fits on one page. Notice that the text at the top of the page is in the left hand corner. This is done using a control command called **`\hfill`**. Notice also that the title line is centered using a control command. The printing of the title in boldface type is achieved by the control command **`\bf`**. The text to be printed for the title is enclosed inside curly brackets. This is the standard way of marking pieces of material for special treatment in **TEX**. Other special type fonts like italics are also governed by control sequences, and the long dash between numbers such as 1.1–1.7 (an *en dash*) is generated by typing two hyphens in succession. Note also in the paragraph preceding "EXAMPLE" that a minus sign is also represented by an en dash; this is automatic when one puts an expression in the so-called *mathematical mode* that is used to describe the setting of mathematical expressions. Instructions of this sort are enclosed by dollar signs, and for –1.0 the corresponding input is **`$-1.0$`**.

Note that the linear space between different items on the sheet is controlled by the command **`\bigskip`**; there are analogous commands called **`\medskip`** and **`\smallskip`** as well as other spacing controls. If you simply leave a blank line between two lines of text, there will be no extra spacing but there will be a line break in the output and the next line will be indented.

Two basic examples of mathematical format are the fractions in the paragraph titled EXAMPLE" and the displayed formula at the end of this paragraph.

On a one page document it is often distracting to see a page number at the bottom, and the **`\nopagenumbers`** command removes this number from the output.

We have already noted that dollar signs are used to indicate that instructions are being given in the mathematical mode. Because of this, one will not get a dollar sign in the output by simply typing **`$`**. One needs to put a backslash in front of the dollar sign to get a dollar sign to print out on the final document. Similar considerations apply to some other special characters including the percentage sign **`%`**. You should check the list of such special characters and familiarize yourself with them in order to avoid problems with your **TEX** files.

DOCUMENT B. The first line of the file sets the type size slightly larger than the default size. Perhaps the most common way of magnifying the text size is to increase sizes by 20 percent; to do this one puts the command line **`\magnification = 1200`** as the first line of the file. This is equivalent to the command **`\magnification = \magstep 1`** in the introductions to that are stored in the course directory. However, in Document B a slightly smaller magnification factor is better, and this is done with the command **`\magnification = 1095`** which is equivalent to the command **`\magnification = \magstephalf.`** Two new features of this document are the use of additional horizontal spacing controls and the introduction of another of the many special characters in **TEX** (the section sign § which is denoted by **`\S`** in **TEX**). There are too many special characters to attempt a discussion of all of them, and the best way to get acquainted with them is to look at a chart displaying them all; the introductions to **TEX** in the course directory contain such charts. Mathematical mode was used to generate italics for expressions such as $7acd$ because it is more concise and easier on the fingers to write **`7$acd$`** or **`$7acd$`** than it is to write **`7{\it acd}`**. Note that numerals in mathematical mode are always set in ordinary roman type unless you force them to look otherwise.

DOCUMENT C. I think it is much easier to grade examinations if there is only one problem per page, so I have designed this with page breaks given by the **`\vfil\eject`** command sequence. Individuals may want variations on the top line for the student's name, maybe with a longer line or with a second blank for the student to indicate a discussion section number. The **`\matrix`** command is used to provide reasonable alignment for the three linear equations on the first page. If one uses the **`\pmatrix`** command then one will obtain a matrix with the usual big parentheses at both ends, but **`\matrix`** by itself does not put anything on the ends. Other end symbols such as long vertical lines or square brackets can also be set; see the online **TEX** documentation for specifics. On page 2, notice that curly brackets have to be preceded by backslashes in the **TEX** file; in **TEX,** one uses curly brackets as control characters to gather things together.

DOCUMENT D. One major reason for including this document is that it contains many of the typesetting features that are needed for calculus courses. For example, there plenty of integral signs, although there are no standard one dimensional definite integrals from a lower limit a to an upper limit b. However, there are inequalities and subscripts, and within the integrals there are spacing commands that are often needed to make things look better. It is very easy for expressions inside an integral sign to look too cramped or too spread out.

The command **`\operatorname`** allows one to set the word "Volume" in roman type within the mathematical mode; if one simply typed the word, it would be set in italics and it might not be clear if this was a word or a collection of mathematical symbols. Notice that the Halmos "square black dot" convention is used to indicate the end of mathematical arguments.

Greek letters appear at various places throughout the document. Note that the small letters are given as control characters in mathematical mode like **`$\alpha$`** while capital letters are given by capitalizing the initial letter of the name as in **`$\Gamma$`**. Variations on standard forms for these letters can be found in the charts of special characters.