ABSTRACT

The ready availability of laser printers and computerized typesetting software tools has opened new horizons for the composition of camera-ready SAS software reports.

One of the most popular and powerful computerized typesetting tools is TeX. By using embedded formatting commands, TeX can readily handle virtually any typesetting requirement, including creation of sophisticated, ruled statistical tables.

This paper describes how SAS software and TeX were used to write teaching evaluation reports for the University of Oregon College of Business. By using TeX and SAS software to produce these reports, the author was able to rapidly condense an existing multipage report into a new, simpler, single-page format that is easy to read, analyze and work with over time.

Use of TeX to format statistical results directly also allows SAS output to be immediately integrated into larger statistical reports (which have also been prepared in TeX) without the need for substantial reformatting, while also eliminating opportunities for transcription errors. Using TeX to format SAS reports also facilitates report maintenance and promotes output device independence.

The author offers tips for those who would like to use TeX to format their own SAS reports and notes some limitations associated with using TeX for SAS report formatting.

INTRODUCTION

In the "bad old days," report output was typically produced on a lineprinter, usually on multipart green bar stock forms (often with uppercase characters only), with less-favored report recipients struggling to read barely legible third or fourth imprint carbon copies.

Today, in the "good new days," report output can be routed to high speed PostScript laser printers or phototypesetters, and there is really no justification for producing ugly, inefficient, and hard to read reports.

Unfortunately, from sheer inertia, many programmers who write custom DATA step reports with SAS are still inflicting nasty, old-fashioned lineprinter-produced reports on their clients.

The intent of this paper is to introduce you to a new tool which will help you break out of those bad old habits when writing custom SAS reports. This new tool is TeX, a computerized typesetting package written by Donald Knuth of Stanford University, which is a world standard for the production of technical documentation by mathematicians, physicists and computer scientists.

Before we begin, please understand—TeX isn’t a what you see is what you get word processor, like Microsoft Word or WordPerfect. Rather, it is a typesetting program which employs a suite of well defined and extensible markup codes to represent special characters and to do document formatting tasks. Some people consider it dauntingly complicated and frustrating, while other people find it elegant and powerful.

Because TeX is so often misunderstood, let me begin by showing you how to use TeX with SAS to write a simple custom DATA step report. After telling you how to learn more about the nuts and bolts of using TeX, I’ll then describe how we use TeX to handle production of a variety of SAS reports, including the production of teaching evaluation reports for the College of Business, production of barcoded check-in rosters for student housing, and production of custom statistical reports.

A SIMPLE NON-\TeX DATA STEP REPORT

The SAS Language and Procedures Guide, Usage, Version 6, First Edition, Volume 1, does an excellent job of showing SAS programmers how to produce a normal (non-\TeX) DATA step reports using FILE and PUT statements (see Part 8, pps. 439-469).

For example, let us assume that we have some hypothetical production quality control data which looks like:

```
A 1 15750 10
A 2 143500 17
A 3 15225 18
B 1 13950 25
B 2 15600 11
B 3 17800 29
B 4 17536 23
```

where the 1st column is a plant code letter, the 2nd column is a lot number, the 3rd column is the size of the lot in units and the 4th column is the number of bad units in that lot.

SAS code to read that little data set and produce a somewhat trivial custom report summarizing it might look something like:

```
data badlots;
  file 'plant.dat';
  input plant $1. lotnum lotsize bad;

data _null_; set badlots end=eof;
  if .ne.1 then do;
    sumsize=sumsize+lotsize;
    sumbad=sumbad+bad;
  end;
  if last.plant then do;
    totsize=sumsize;
    totbad=sumbad;
    put @9 'Plant totals:' @26 sumsize @36 sumbad /:
  end;

if first.plant then do;
  sumsize=0;
  sumbad=0;
  put plant $;
end;

put @9 lotnum &26 lot size @36 bad;
sumsize=lotsize;
sumbad=bad;
if last.plant then do;
  totsize=sumsize;
  totbad=sumbad;
  put @9 'Plant totals:' @26 totsize @35 totbad /:
end;

if eof then do;
  if totsize=0 then do;
    put @9 'G9ARD TOTALS:' @26 totsize @35 totbad /:
  end;
```

\TeX is so often misunderstood, let me begin by showing you how to use \TeX with SAS to write a simple custom DATA step report. After telling you how to learn more about the nuts and bolts of using \TeX, I’ll then describe how we use \TeX to handle production of a variety of SAS reports, including the production of teaching evaluation reports for the College of Business, production of barcoded check-in rosters for student housing, and production of custom statistical reports.

\footnote{SAS is a registered trademark of SAS Institute, Inc.}
Running that code through SAS would produce a report which looks like:

**LOT EXCEPTION REPORT**

<table>
<thead>
<tr>
<th>Plant</th>
<th>Lot</th>
<th>Size</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>16750</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>14300</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>12250</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>45275</td>
<td>45</td>
</tr>
<tr>
<td>Plant totals:</td>
<td></td>
<td>109571</td>
<td>133</td>
</tr>
</tbody>
</table>

Now, of course this trivial little report **could** be produced using any of a number of different 'canned' SAS procedures (such as PROC PRINT, PROC REPORT, or PROC TABULATE), but (as every report writer quickly learns) it seems there are always inescapable reports which can only be readily handled by writing a custom SAS DATA step report. Granted, our current example might be trivially simple, but the basic ideas employed in TeXifying even that minimal little report will scale directly to far larger and more convoluted reporting projects.

**TeXifying Our Sample Report**

We begin by making some decisions about our general TeX environment (i.e., the design of our page).

**Page Design of Our Sample Report**

Because TeX developed out of a typesetting paradigm, we have excellent control over global document design characteristics such as page size, page numbering, margin size, type family, type size, leading, paragraph formatting, etc. However, this also means that we need to be relatively specific about what we want. For this example, let us assume that we are going to use:

- a standard 8.5" by 11" page in portrait orientation (i.e., the design of our page).
- a 1.25" left margin, and 3/4" margins on the top, right and bottom,
- ragged right lines (each line is NOT filled with additional space as it would need to in order to create a flush right margin),
- blocked format (no indentation of new 'paragraphs'),
- no page numbers,
- a single typeface (ten point Computer Modern Roman),
- a standard 8.5" by 11" page in portrait orientation (i.e., the design of our page).

The TeX commands needed to get that page setup look like:

```
\textwidth=6.5in \textheight=9.5in \hoffset=-.25in \topmargin=-.25in \vsize=9.5in \voffset=-.25in
\hsize=6.5in \textwidth=6.5in \textheight=9.5in \hoffset=.25in \vsize=9.5in \voffset=-.25in
```

Notice that each TeX command begins with a backslash (\) and that virtually all TeX commands need to be written in lowercase only (TeX commands are case sensitive, and thus trying to use the (wrong) command \TOPPAGENUMBERS instead of \nopagenumbers would result in an error.)

Also notice that TeX commands which refer to dimensions always include a unit of measure, rather than just a 'raw' number. TeX makes no assumptions about whether you are working in inches or centimeters, or printer's pica, points or ems. Rather, TeX requires you to explicitly specify the units you are using for each dimension, and TeX allows you to freely mix various dimensions throughout your document, using whichever one best meets your requirements at a given time.

A third thing to notice in looking at these first initial commands is that any TeX command can be followed by comments, offset from the body of the command by a percent sign (%). Since even the most dedicated TeXophile will admit that some TeX commands can be rather cryptic, it is a good idea to extensively comment any file containing TeX commands, just so you would know any SAS or C program.

Having established our general page format, let's proceed to manually produce a TeX version of the body of our prototype report. Once we have a manually-written version which works cleanly, we'll then design SAS statements to automatically generate that TeX code on a production basis.

**Continuing to Craft Our Prototype TeX-fied Report**

First comes the title for the report:

```
\bf \texttt{LOT EXCEPTION REPORT}
```

The \bf command used while defining the title means that the title will be displayed in a bold face version of the current font. Bolding is limited to the text that's within the curly brace delimiters.

Next, we'll establish 'tab stops' for use in constructing the main body of the table. Because TeX uses proportional (non-fixed-width) fonts, we can't simply 'space over' space by space to where we might want output to be displayed -- in TeX every character takes up a different amount of space depending upon its natural width. The proper way to insure that our output will line up nicely in columns is to establish 'tab stops,' and then use TeX \tab to align our output.

For now, let's set up tab stops for our real columns interleaved with dummy columns each two em's wide. By including extra dummy columns, we can readily adjust the intercolumn spacing by changing the width of the dummy columns, should we later need to do so.

```
|\settabs| \\hglue 2em \hglue 0.3\in \hglue 0.75\in \hglue 0.3\in \hglue 0.3\in
|       | \hglue 0.3\in \hglue 0.75\in \hglue 0.3\in \hglue 0.3\in | \cr
```

Notice that because our \settabs command was too wide to fit on a single line, we entered it across several lines. This can generally be done with any TeX statement, so long as you don't split a line in the middle of a TeX command token. No continuation line 'continuation character' is required to signal the existence of a continuation line, nor do you need to indent continuation lines (although indented continuation lines certainly increase the readability of TeX code).

Also, understand that while we've used \hglue as an easy way to specify the width of our columns, we could just as readily have put in any piece of text, or even just a series of X's, to show
\texttt{\LaTeX} the spacing between tab stops we desire. All \texttt{\LaTeX} sees is the width of the 'stuff' that is between successive ampersands on the \texttt{\settabs} command.

After we've gotten our tab stops set up, our column headings are easy to generate:

\begin{verbatim}
\settabs \& \{bf Plant\} \& \{bf Lot\} \& \{bf Size\} \\
\{bf Bad\} \cr
\end{verbatim}

Some notes about our column headings:

- Tabbed lines in \texttt{\LaTeX} must always begin with the literal character string \texttt{\&} and must always end with the literal character string \texttt{\cr}. One logical tabbed line can extend over multiple physical lines in your file - it doesn't all have to fit on a single physical line.

- Whenever \texttt{\LaTeX} comes to an ampersand (\&), \texttt{\LaTeX} always goes to the 'next' tabstop you've defined even if the 'next' tabstop is to the left of your current position! \texttt{\LaTeX} keeps track of the 'next' tabstop simply by counting how many ampersands it has already seen on the current tabbed line, rather than by figuring out where the next tabstop to the 'right' of the current position might be (the way an old-fashioned mechanical typewriter might).

- Two successive ampersands (\&\&) are used to skip over the dummy columns that we included for spacing purposes.

- The spaces which are present in the \texttt{\LaTeX} code (except those following the \texttt{\&}) command are there for ease of reading, not because they need to be present for \texttt{\LaTeX} to process the line. The spaces should probably be removed from any \texttt{\LaTeX} code actually used to typeset this table on a production basis to prevent \texttt{\LaTeX} from typesetting 'extra' spaces around headings:

\begin{verbatim}
\settabs \& \{bf Plant\} \& \{bf Lot\} \& \{bf Size\} \\
\{bf Bad\} \cr
\end{verbatim}

The horizontal rule (or 'hrule') separating the heading from the detail lines is added with the commands:

\begin{verbatim}
\hrule width 3 in height 0.5 pt
\end{verbatim}

The \texttt{\hrule} which brackets the \texttt{\hrule} are present to insure that there is some space on either side of that horizontal line. (A \texttt{\hrule} is about the tallness of a single line of text, so doing a single \texttt{\hrule} is essentially equivalent to skipping a single line of text).

The detail lines follow basically the same pattern as the column headings:

\begin{verbatim}
\& A \& 1 \& 15750 \& 10 \cr
\& 2 \& 14300 \& 17 \cr
\& 3 \& 15275 \& 18 \cr
\savekip \& \{bf Plant Totals:} \& \$45275 \& \$45 \cr
\restorekip \& \{bf Bad\} \cr
\end{verbatim}

The \texttt{\savekip} which bracket the \texttt{\restorekip} are present to insure that there is some space on either side of that horizontal line. (A \texttt{\savekip} is about the tallness of a single line of text, so doing a single \texttt{\savekip} is essentially equivalent to skipping a single line of text).

The detail lines follow basically the same pattern as the column headings:

\begin{verbatim}
\& A \& 1 \& 15750 \& 10 \cr
\& 2 \& 14300 \& 17 \cr
\& 3 \& 15275 \& 18 \cr
\savekip \& \{bf Plant Totals:} \& \$45275 \& \$45 \cr
\restorekip \& \{bf Bad\} \cr
\end{verbatim}

Virtually all of the \texttt{\LaTeX} commands used in typesetting these detail lines are ones which you saw before in the column headings, with the exception of a couple of \texttt{\smallskip} lines added for esthetic reasons. (As a general rule of thumb, four \texttt{\smallskip} equal two \texttt{\medskip} which in turn equals one \texttt{\bigskip}).

Some notes about our column headings:

- Tabbed lines in \texttt{\LaTeX} must always begin with the literal character string \texttt{\&} and must always end with the literal character string \texttt{\cr}. One logical tabbed line can extend over multiple physical lines in your file - it doesn't all have to fit on a single physical line.

- Whenever \texttt{\LaTeX} comes to an ampersand (\&), \texttt{\LaTeX} always goes to the 'next' tabstop you've defined even if the 'next' tabstop is to the left of your current position! \texttt{\LaTeX} keeps track of the 'next' tabstop simply by counting how many ampersands it has already seen on the current tabbed line, rather than by figuring out where the next tabstop to the 'right' of the current position might be (the way an old-fashioned mechanical typewriter might).

- Two successive ampersands (\&\&) are used to skip over the dummy columns that we included for spacing purposes.

- The spaces which are present in the \texttt{\LaTeX} code (except those following the \texttt{\&}) command are there for ease of reading, not because they need to be present for \texttt{\LaTeX} to process the line. The spaces should probably be removed from any \texttt{\LaTeX} code actually used to typeset this table on a production basis to prevent \texttt{\LaTeX} from typesetting 'extra' spaces around headings:

\begin{verbatim}
\settabs \& \{bf Plant\} \& \{bf Lot\} \& \{bf Size\} \\
\{bf Bad\} \cr
\end{verbatim}

The horizontal rule (or 'hrule') separating the heading from the detail lines is added with the commands:

\begin{verbatim}
\hrule width 3 in height 0.5 pt
\end{verbatim}

The \texttt{\hrule} which brackets the \texttt{\hrule} are present to insure that there is some space on either side of that horizontal line. (A \texttt{\hrule} is about the tallness of a single line of text, so doing a single \texttt{\hrule} is essentially equivalent to skipping a single line of text).

The detail lines follow basically the same pattern as the column headings:

\begin{verbatim}
\& A \& 1 \& 15750 \& 10 \cr
\& 2 \& 14300 \& 17 \cr
\& 3 \& 15275 \& 18 \cr
\savekip \& \{bf Plant Totals:} \& \$45275 \& \$45 \cr
\restorekip \& \{bf Bad\} \cr
\end{verbatim}

Sharp-eyed readers will also notice that the rows of dashes which were used in the lineprinter version of our report have been deleted from the \texttt{\LaTeX} version. Generally speaking, selective use of bolding or italics is the preferred way of drawing the reader's attention to totals or headings in a \texttt{\LaTeX} report.

Our last task is to signal \texttt{\LaTeX} that we're done:

\begin{verbatim}
\bye
\end{verbatim}

Our final manually constructed \texttt{\LaTeX}-ified report thus looks like:

\begin{verbatim}
\settabs \& \{bf Plant\} \& \{bf Lot\} \& \{bf Size\} \\
\{bf Bad\} \cr
\end{verbatim}

RUNNING OUR LITTLE REPORT THROUGH \texttt{\LaTeX}

O.K. - We've now written all the \texttt{\LaTeX} commands we need to make our report, and we'll assume that they're available in a text file on our computer called myreport.\texttt{\LaTeX} ... but how do we process that file into something printable on a laserprinter?

\texttt{\LaTeX} is typically run using a command like:

\texttt{tex myreport.\texttt{\LaTeX}}

If no fatal errors are found, this should generate a \texttt{.dvi} (device independent format) file. If \texttt{\LaTeX} does detect errors, it will report them and allow you to correct them.

Your \texttt{.dvi} file can then be previewed on a workstation or microcomputer. For instance, here at the University of Oregon, I preview \texttt{.dvi} files on a NeXT workstation using \texttt{\TeX}View, but \texttt{X}11-based previewers are also popular, as are previewers which run on PC's and Macintoshes.

When your file is finally typeset to your satisfaction, you then push the \texttt{.dvi} file through a 'filter' (or 'driver') program to
generate output suitable for printing on a particular type of laser printer or phototypesetter. I use and strongly endorse Tom Rokicki's dvips program for PostScript output. To run it on a .dvi file, simply say:

dvips myreport.dvi

Other commonly used .dvi converters include Nelson Beebe's drivers for the HP Laser Jet (and Laser Jet Plus) printers, and various drivers which are available for microcomputer dot matrix printers.

You can even convert .dvi files back into regular ASCII text for output to a traditional lineprinter, but the result of such a conversion (via a program such as crudetype) can be rather unsatisfactory due to the limited graphic capabilities of vanilla ASCII output devices.

In any event, the actual \TeX{} output from our manually \TeX{}-ified prototype report looks like:

LOT EXCEPTION REPORT

<table>
<thead>
<tr>
<th>Plant</th>
<th>Lot</th>
<th>Size</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>15750</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>14300</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>15225</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>45275</td>
<td>45</td>
</tr>
</tbody>
</table>

B

|     | 1   | 13960  | 25  |
|     | 2   | 15000  | 11  |
|     | 3   | 17800  | 29  |
|     | 4   | 17536  | 23  |
|     |     | 64296  | 88  |

Grand Totals: 109571 133

WRITING \TeX{} CODE FROM SAS

We're now ready to begin generating \TeX{} code automatically from within SAS. Essentially all we need to do is modify our PUT commands to include the required \TeX{} directives. Done in as straightforward a way as possible, that results in a SAS program which looks like:

data badlots;
  infile 'plant.dat';
  input plant $1. lotnum lotsize bad;
  data small;
  set badlots end=.eof;
  by plant;
  file 'rep.tex';
  if _n_ then do;
    put '{\bf LOT EXCEPTION REPORT}';
    put '{\bf Plant};
    put '{\bf Lot}';
    put '{\bf Size}';
    put '{\bf Bad}';
    put '
    put '{\bf Plant Totals:}';
    put '{\bf Sum Size}';
    put '{\bf Sum Bad}';
    put '
    end;
    if first.plant then do;
      sumsize=0;
      sumbad=0;
      put plant $;
      end;
    put '{\bf Lot:}';
    put '{\bf Lotnum}';
    put '{\bf Lotsize}';
    put '{\bf Bad}';
    put '
    put '{\bf Plant Totals:}';
    put '{\bf Sum Size}';
    put '{\bf Sum Bad}';
    put '
    end;
    if last.plant then do;
      totsize=sumsize;
      totbad=sumbad;
      put '{\bf Plant Totals:}';
      put '{\bf Sum Size}';
      put '{\bf Sum Bad}';
      put '
      end;
    if eof then do;
      put '{\bf Grand Totals:}';
      put '{\bf Tot Size}';
      put '{\bf Tot Bad}';
      put '{\bf bye}';
    end;
  end;
This same basic approach will work for virtually any custom \TeX{}-ified report.

You now know enough to be able to generate many basic \TeX{}-ified reports (and a lot of not-so-simple \TeX{}-ified reports) on your own.

At this point, let's go on to talk about some of the SAS-produced \TeX{} reports we've actually generated here at the University of Oregon.

AN ACTUAL EXAMPLE: EVALUATIONS

For many years now, the University of Oregon has employed student evaluations of teaching as one important component of its overall promotion and tenure assessment program.

Near the end of each term, students are asked to anonymously assess their instructor's performance during the term by filling out a multiple-choice teaching evaluation. The completed questionnaires are collected by a member of the class, scanned and processed by the Computing Center and distributed by the CBA at the start of the next term to the instructors and to administrators; public copies are made available for student review in the library and in student government offices.

A year or two ago, the University of Oregon College of Business Administration (CBA) revamped the teaching evaluation
As a Description of This Course/Instructor, This Statement Is:

1. Learning: You found the course intellectually challenging and stimulating...
2. You have learned something that you consider valuable...
3. Your interest in the subject has increased as a consequence of this course...
4. 1...
5. Enthusiasm: Instructor was enthusiastic about teaching the course...
6. Instructor was dynamic and energetic in conducting the course...
7. Instructor maintained a balance between lectures and discussions...
8. Instructor's style of presentation held your interest during class...
9. Organization: Instructor's explanations were clear...
10. Course materials were well prepared and clearly explained...
11. Course objectives were clearly stated so that you knew what was expected...
12. Instructor gave feedback that facilitated learning...
13. Group Interaction: Students were encouraged to participate in class discussions...
14. Students were engaged in class discussions...
15. Students were encouraged to share their ideas and knowledge...
16. Students were assigned to small groups for class projects and given meaningful assignments...
17. Individual Rappier: Instructor was friendly towards individual students...
18. Instructor made students feel welcome in seeking help with or outside of class...
19. Instructor had a genuine interest in individual students...
20. Instructor was adequately accessible to students during office hours or by email...
21. Instructor's knowledge of the subject was vast...
22. Instructor was well prepared for each class session...
23. Instructor presented the background or origin of ideas/concepts developed in class...
24. Instructor presented points of view that were clear and accurate...
25. Instructor addressed issues in a timely and appropriate manner...
26. Instructor's style of teaching was interesting...
27. Instructor's grading was fair and appropriate...
28. Instructor's teaching methods were...
questionnaire it used. Students and faculty alike had been unhappy with the existing questionnaire, and had expressed dissatisfaction with the hard-to-read multi-page summary reports produced from the evaluations. A CBA committee designed a new questionnaire, constructed a custom mark-sense form for the questionnaire, and then approached the Computing Center for assistance in developing suitable new reports.

Important objectives in developing that new report included:

- Accommodating a near doubling of questionnaire items (from 22 items to 41); and
- Producing the required report in a single page format (instead of the old multi-page format) while using normal 8.5x11" forms for ease of filing and duplication (instead of traditional 11x14 7/8" green bar forms).

While it might have been physically possible to fit all the required output on a single 8.5x11" lineprinter page, the result would have been confusing and unusable.

By using TeX code written from SAS, however, we got the sort of clean solution the College of Business was looking for.

TeX has small (but legible) fonts. TeX has excellent table handling capabilities (shown, but not discussed in this paper) which made laying out the report quite an easy task. Bolding could be used where needed for emphasis, and columns could be grouped together through use of a lined tabular format. TeX even made it easy to write a companion explanatory document (including all required statistical formulas).

The approach we followed was essentially the same as that described in our initial example: a dummy prototype report was manually developed and approved by the College of Business, and then SAS code was written to automate the production of the report. This particular application required about a thousand lines of SAS code (mainly consisting of DATA step statements).

Composing TeX Commands Prior to PUT-ing Them

Our SAS code was somewhat longer than we'd anticipated because we encountered an interesting problem which required us to alter the way we'd planned to compose TeX commands for output from SAS.

The problem related to SAS's trim(var) and concatenation functions. We'd wanted to trim and concatenate some string variables prior to PUT-ing them to a file, but unfortunately the trim function couldn't be used directly in a PUT statement. For example:

```
put '{\bf INSTUCTOR:} \quad \texttt{\quad trim(lastname) }\quadtrim(firstname); 
```

The solution to this problem was to use the trim function and the concatenate operator ([]) to build our TeX commands in a long temporary string variable, finally PUT-ing that temporary string variable once it had been completely composed:

```
length instname $60 temp $200;
instname=trim(lastname)]; 
|[trim(firstname); put temp;
```

This worked, but was rather less elegant than we might have liked. We're still looking for an alternative cleaner solution.

Running the CBA Report

Output from this application for a typically quarter consists of summaries for about a hundred course sections. When we began to run the report on a production basis, it ran quite smoothly except for an interesting and unanticipated problem: because the report involved reading and printing character strings (such as instructor names and course titles), we inadvertently discovered that instructor names and course titles occasionally contained certain special characters which TeX would 'misinterpret' unless special precautions were taken. For example, numerous course titles included ampersands ("Marketing Concepts & Theory") — those ampersands which would be 'misinterpreted' by TeX as a tab character (rather than being treated as a literal ampersand) unless 'escaped' with a backslash prefix.

In general, however, the new report came up exceptionally easily and was accepted with great enthusiasm by the client. See the figures shown below.

ANOTHER EXAMPLE: MAKING BAR CODES

Buoyed by the success of our experiment with the CBA's evaluations, we proceeded to prototype a SAS application to generate barcoded dormitory check-in rosters and barcoded laser printed gummed labels for use by the University's Housing Department.

One of TeX's greatest strengths is its ability to employ any of a huge variety of different fonts, and to even permit users to create new fonts as required using an auxiliary program called METAFOFT.

In our case, Dimitri Vulis (CDLWBEASAGISS) had already prepared and distributed METAFOFT source for 3 of 9 barcodes, based on a BASIC language barcoding program originally written by Bill Wood of Milwaukee, WI, and Bill Simon of Enfield, CT. As far as TeX is concerned, barcoded letters are just another font, and thus are trivial to use.

The programmer for this task was someone who was well familiar with SAS, but completely new to TeX. After helping the programmer prepare a prototype page of the required report, she was readily able to implement the report in SAS with only a few minor problems stemming from our collective lack of barcode production experience.

For example, neither of us knew that code 39 barcodes had to start and end with an asterisk, nor did we know that it was desirable to leave a grace margin around the actual bar code. These minor changes were easy enough to implement.

A more subtle problem arose when the programmer embarked on generating barcoded gummed labels from a SAS DATA step. A sample page was generated and the spacing between the lines tweaked by trial and error to insure proper vertical alignment on the gummed label stock we'd be using.

A day or so later, the programmer returned, quite frustrated to find that the trial run had noticeable vertical alignment problems! After some head scratching, we finally noticed a subtle change: it turned out that the programmer had decided to add a comma between each person's capitalized first name and capitalized last name. TeX automatically increased the spacing between lines to reflect the descenders on the commas, thereby resulting in a small but significant increase in line spacing, which accumulated to a noticeable vertical spacing error by the bottom of the page.

Omission of the commas eliminated the problem (we could also have re-tweaked the default baselinekip). Remember, in TeX, as in all typesetting, line spacing is infinitely adjustable, and there is no concept of 'single spacing' or 'double spacing' per se.

A FINAL EXAMPLE: STATISTICAL TABLES

There are numerous additional instances when being able to generate TeX output directly from SAS is quite handy.
<table>
<thead>
<tr>
<th>NAME</th>
<th>SSN</th>
<th>HALL</th>
<th>RM</th>
<th>BOX</th>
<th>PHONE</th>
<th>SP</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANDERSON S</td>
<td></td>
<td>MCR</td>
<td>405</td>
<td>63974</td>
<td>346-1234</td>
<td></td>
</tr>
<tr>
<td>BIRCHWOOD TOM K</td>
<td></td>
<td>MCR</td>
<td>402</td>
<td>63509</td>
<td>346-2345</td>
<td>LW</td>
</tr>
<tr>
<td>BROWN MARTHA</td>
<td></td>
<td>MCR</td>
<td>206</td>
<td>63203</td>
<td>346-3456</td>
<td></td>
</tr>
<tr>
<td>BRUNO ICHIRO</td>
<td></td>
<td>MCR</td>
<td>205</td>
<td>64118</td>
<td>346-4567</td>
<td></td>
</tr>
<tr>
<td>CHESTER CLYDE</td>
<td></td>
<td>MCR</td>
<td>103</td>
<td>63915</td>
<td>346-5678</td>
<td>LW</td>
</tr>
<tr>
<td>CHOWN CLIFFORD</td>
<td></td>
<td>MCR</td>
<td>305</td>
<td>63755</td>
<td>346-6789</td>
<td></td>
</tr>
<tr>
<td>CHUN-WING CHAN</td>
<td></td>
<td>MCR</td>
<td>414</td>
<td>63788</td>
<td>346-7890</td>
<td></td>
</tr>
<tr>
<td>CRAMER R G</td>
<td></td>
<td>MCR</td>
<td>404</td>
<td>63096</td>
<td>346-8901</td>
<td></td>
</tr>
<tr>
<td>CREST LISA</td>
<td></td>
<td>MCR</td>
<td>314</td>
<td>63746</td>
<td>346-9012</td>
<td>LW</td>
</tr>
</tbody>
</table>
For example, when you are generating complicated statistical tables it is very easy to have typographical errors sneak in during the transcription process en route to the final printed report. If you can generate a camera-ready table, the opportunity for those transcription errors is eliminated.

Consider the two 'handgun stopping power' statistical tables shown below. Both of these tables were produced by writing \TeX code directly from SAS.

It is a virtual certainty that manually typesetting those tables would have been highly susceptible to error. In particular, it would be quite hard to accurately transcribe table 2 with its columns after columns of pluses, minuses and spaces. By writing \TeX code directly from SAS, there's no need to do that sort of transcription.

Also note that because \TeX has excellent facilities for generating equations and greek letters, statistical information could be appended right to the bottom of the report, making it self-documenting and allowing it to be largely independent of explanatory documents.

Another advantage of using \TeX and SAS to produce statistical tables is that \TeX can be made to dynamically determine the required columnar widths (through use of \halign-type tables), thereby automatically adjusting the width of each column as may be required to accommodate that column's contents. Thus, for instance, if the 'Misc' column in table one needed to be widened to accommodate a longer entry, \TeX would automatically make that adjustment since this table was set using \halign rather than fixed tabstops.

That sort of automatic adjustment greatly reduces a programmer's report maintenance work load.

NURTURING YOUR NASCENT \TeX SKILLS

While we've now shown you just enough \TeX to 'make you dangerous,' and a few real examples to whet your appetite, you should really learn more about \TeX before beginning to use it for your own SAS reporting requirements.

The \TeXbook; \TeX For The Impatient

One excellent way to learn about \TeX is to work your way through Donald Knuth's \TeXbook, published by Addison Wesley, Reading MA, January 1986, ISBN 0-201-13448-9. This is the reference for \TeX, and it contains numerous end-of-chapter exercises designed to enhance your \TeX knowledge. However, beware, \TeX as presented by Knuth has a steep learning curve. Working through his book will take some time and may give you headaches. A more approachable alternative \TeX book is Paul Abrahams' \TeX For the Impatient, Addison Wesley, Reading MA, 1990, ISBN 0-201-51375-7. Although not as authoritative or detailed as Knuth's book, it is generally well received by beginning \TeX users.

Free \TeX Manuals

Those of you who are on the Internet can also retrieve two free manuals designed to help you get started with \TeX.

The older of those manuals is called A Gentle Introduction to \TeX; A Manual for Self Study, by Michael Doeb of Manitoba. His manual is available free via anonymous ftp from MIORD.SHSU.EDU in directory GENTLE. If you have only e-mail access to the Internet (and cannot use anonymous FTP instead), send a mail message consisting of the text SEEME GENTLE to the address FIESERVER@SHSU.EDU.

The other free introduction to \TeX available to Internet users is my own Using \TeX on the VAX to Typeset Documents: A Primer. You can obtain a free copy of that document via anonymous FTP from dace@wesonex.edu (128.233.32.191) (get the .ps files from the directory /ftp/pub/tex/processed/). If you have only e-mail access, get a copy of my manual by sending the message SEEME TEX.PRIMER to FIESERVER@SHSU.EDU.

Notwithstanding the somewhat misleading title of my primer, the majority of that book is a general introduction to writing \TeX suitable for virtually any system.

\TeX Users Group

Another excellent source of information is the \TeX Users Group (TUG), PO Box 9506, Providence, RI 02940 (Phone: 401-751-7790, FAX: 401-751-1071). When you join TUG, you get a whole host of benefits, including a subscription to TUGboat (the \TeX User Group's newsletter), information on \TeX training opportunities, \TeX meeting announcements, a directory of TUG members, access to software, etc.

comp.text.tex

Once you've been working with \TeX for a while, you may also want to begin following comp.text.tex, a USENET newsgroup devoted to \TeX topics. If you have a truly tough or obscure \TeX question, comp.text.tex will put you in touch with the worldwide \TeX community, and someone may help you out with your problem.

However, before posting a query to comp.text.tex, sure to first exhaust alternative local resources, and also be sure to check the answers included in the frequently asked questions (FAQ's), "posted" to comp.text.tex each month before posting your question — your question may not be as unique (or insoluble) as you think it is! Another very informative periodic posting in comp.text.tex is entitled 'Supplementary \TeX Information' and you should probably review it before posting a query, too.

SUMMARY

You've now had a whirlwind introduction to using \TeX with SAS to format custom data step reports. You should be able to produce at least simple \TeX reports from SAS with minimal problems, and you have a general idea of \TeX's potential for enhancing SAS output. You also know where to get additional information about using \TeX.

There are a lot of \TeX and dvips topics we haven't talked about, including use of PostScript fonts (such as Time Roman or Helvetica), support for PostScript graphics, the ability to generate overprint screens on each page (i.e., faint outlines stating 'DRAFT' or 'CONFIDENTIAL', etc.); \TeX's ability to handle foreign languages (including even Kanji and Arabic), generation of tables of contents and indices, handling block indentations and bulleted lists, etc. Nonetheless, you have at least seen the 'tip of the \TeX iceberg' in this paper. Once you try \TeX yourself, I think you'll agree that \TeX can multiply the power of the SAS system manyfold.

SOURCES FOR \TeX SOFTWARE

One of the most daunting things which keeps people from getting started with \TeX is getting \TeX and its support programs in the first place. One easy solution to this problem is to buy a NeXT workstation, since \TeX, dvips and TeXview (a \TeX previewer) all come bundled free with that machine!

If you already have some other UNIX box, you can obtain a UNIX version of \TeX via anonymous FTP over the Internet from byron.u.washington.edu in directory pub/unix/tex. A UNIX version is also available on tape for a fee by calling the University of Washington at 206-543-0220.

A VAX/VMS version of \TeX is at ymir.claremont.edu in direct-
Table 1. ‘One shot stop’ performance of selected handgun cartridges.

<table>
<thead>
<tr>
<th>Rank</th>
<th>Tot</th>
<th>% Stop</th>
<th>Cartridge</th>
<th>Mun</th>
<th>Wgt</th>
<th>Type</th>
<th>+P?</th>
<th>Misc</th>
<th>Velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>482</td>
<td>65.97</td>
<td>.357 Magnum</td>
<td>Fed</td>
<td>125</td>
<td>JHP</td>
<td></td>
<td></td>
<td>1455</td>
</tr>
<tr>
<td>2</td>
<td>701</td>
<td>80.83</td>
<td>.38 Special</td>
<td>Fed</td>
<td>153</td>
<td>SJHP</td>
<td></td>
<td></td>
<td>1105</td>
</tr>
<tr>
<td>3</td>
<td>603</td>
<td>81.05</td>
<td>.45 Auto</td>
<td>Fed</td>
<td>105</td>
<td>JHP</td>
<td></td>
<td></td>
<td>1260</td>
</tr>
<tr>
<td>4</td>
<td>591</td>
<td>81.38</td>
<td>.45 Auto 100</td>
<td>Fed</td>
<td>115</td>
<td>JHP</td>
<td></td>
<td></td>
<td>1260</td>
</tr>
<tr>
<td>5</td>
<td>110</td>
<td>82.05</td>
<td>.44 Magnum</td>
<td>Fed</td>
<td>125</td>
<td>SJAP</td>
<td></td>
<td></td>
<td>1200</td>
</tr>
<tr>
<td>6</td>
<td>751</td>
<td>82.16</td>
<td>.45 Auto R·P</td>
<td>Fed</td>
<td>115</td>
<td>JHP</td>
<td></td>
<td></td>
<td>1290</td>
</tr>
<tr>
<td>7</td>
<td>369</td>
<td>82.18</td>
<td>.41 Magnum</td>
<td>Fed</td>
<td>115</td>
<td>JHP</td>
<td></td>
<td></td>
<td>1290</td>
</tr>
<tr>
<td>8</td>
<td>354</td>
<td>82.61</td>
<td>.45 Auto</td>
<td>Fed</td>
<td>115</td>
<td>JHP</td>
<td></td>
<td></td>
<td>1290</td>
</tr>
<tr>
<td>9</td>
<td>321</td>
<td>82.96</td>
<td>.44 Special 100</td>
<td>Fed</td>
<td>115</td>
<td>JHP</td>
<td></td>
<td></td>
<td>1290</td>
</tr>
<tr>
<td>10</td>
<td>309</td>
<td>83.23</td>
<td>.44 Special 125</td>
<td>Fed</td>
<td>115</td>
<td>JHP</td>
<td></td>
<td></td>
<td>1290</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Total</td>
<td>Fed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1290</td>
</tr>
</tbody>
</table>

Note: read across each row. The apparent performance of a given row's cartridge is statistically superior to those alternatives marked with a plus sign. The apparent performance of a given row's cartridge is statistically inferior to those alternatives marked with a minus sign. A blank indicates that insufficient evidence exists for us to find a statistically significant positive or negative difference.


Table 2. Comparative stopping power of selected handgun cartridges (test: equality of proportions, $\alpha = 0.05$).

<table>
<thead>
<tr>
<th>Rank</th>
<th>Tot</th>
<th>% Stop</th>
<th>Cartridge</th>
<th>Mun</th>
<th>Wgt</th>
<th>Type</th>
<th>+P?</th>
<th>Misc</th>
<th>Velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>482</td>
<td>65.97</td>
<td>.357 Magnum</td>
<td>Fed</td>
<td>125</td>
<td>JHP</td>
<td></td>
<td></td>
<td>1455</td>
</tr>
<tr>
<td>2</td>
<td>701</td>
<td>80.83</td>
<td>.38 Special</td>
<td>Fed</td>
<td>153</td>
<td>SJHP</td>
<td></td>
<td></td>
<td>1105</td>
</tr>
<tr>
<td>3</td>
<td>603</td>
<td>81.05</td>
<td>.45 Auto</td>
<td>Fed</td>
<td>105</td>
<td>JHP</td>
<td></td>
<td></td>
<td>1260</td>
</tr>
<tr>
<td>4</td>
<td>591</td>
<td>81.38</td>
<td>.45 Auto 100</td>
<td>Fed</td>
<td>115</td>
<td>JHP</td>
<td></td>
<td></td>
<td>1260</td>
</tr>
<tr>
<td>5</td>
<td>110</td>
<td>82.05</td>
<td>.44 Magnum</td>
<td>Fed</td>
<td>125</td>
<td>SJAP</td>
<td></td>
<td></td>
<td>1200</td>
</tr>
<tr>
<td>6</td>
<td>751</td>
<td>82.16</td>
<td>.45 Auto R·P</td>
<td>Fed</td>
<td>115</td>
<td>JHP</td>
<td></td>
<td></td>
<td>1290</td>
</tr>
<tr>
<td>7</td>
<td>369</td>
<td>82.61</td>
<td>.41 Magnum</td>
<td>Fed</td>
<td>115</td>
<td>JHP</td>
<td></td>
<td></td>
<td>1290</td>
</tr>
<tr>
<td>8</td>
<td>354</td>
<td>82.96</td>
<td>.45 Auto</td>
<td>Fed</td>
<td>115</td>
<td>JHP</td>
<td></td>
<td></td>
<td>1290</td>
</tr>
<tr>
<td>9</td>
<td>321</td>
<td>83.23</td>
<td>.44 Special 100</td>
<td>Fed</td>
<td>115</td>
<td>JHP</td>
<td></td>
<td></td>
<td>1290</td>
</tr>
<tr>
<td>10</td>
<td>309</td>
<td>83.23</td>
<td>.44 Special 125</td>
<td>Fed</td>
<td>115</td>
<td>JHP</td>
<td></td>
<td></td>
<td>1290</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Total</td>
<td>Fed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1290</td>
</tr>
</tbody>
</table>

Note: read across each row. The apparent performance of a given row's cartridge is statistically superior to those alternatives marked with a plus sign. The apparent performance of a given row's cartridge is statistically inferior to those alternatives marked with a minus sign. A blank indicates that insufficient evidence exists for us to find a statistically significant positive or negative difference.

Some basic information about the test for equality of proportions ...

Hypothesis: $H_0: \pi_1 = \pi_2 = \cdots = \pi_s = 0.5$ vs. $H_1: \pi_1, \pi_2, \cdots, \pi_s \neq 0$

Test statistic: $z = \sqrt{\frac{S_n}{\sqrt{np(1-p)}}}$

where $p_0$ is the proportion of successes in sample $i$, $D_i$ is the hypothesized difference (i.e., zero), and $n_i$ is the size of sample $i$.

Required Assumptions: (a) Independent random samples, and (b) $n_i, n_j$ are sufficiently large for the sampling distributions of $p_i - p_j$ to be approximated by a normal distribution by the CLT. Thus, typically, $|\bar{x}_i - \bar{x}_j|$ should be less than $2\sigma$. See, for example, Larry R. Hunter, Business Statistics: Why and How, 2nd Edition, McGraw Hill, 1983, pp. 802-809.
tory [\text{tex.exe}], or on tape through DECUS. A supported commercial version of \TeX is available for VMS which we’ve had good success with is available from Northlake Software, 812 SW Washington, Suite 1100, Portland, OR 97205, phone: 503-228-5662.

A PC version of \TeX (EmTeX) is available via anonymous FTP from ymir.claremont.edu in directory [.tex.ibm.pc.emtex], on diskette through TUG.

A Macintosh version of \TeX called \text{O\TeX} is available from midway.uchicago.edu, complete with a previewer and a PostScript driver. It is also available on diskette through TUG.

dvips is available from neon.stanford.edu, or via TUG.

x\text{dvi}, an X Window System-based \text{dvi} previewer, is available from export.lcs.mit.edu and is usually a part of most X11 distribution tapes.

The SAS and \TeX examples used in this paper are available via anonymous FTP from decoy.cc.uoregon.edu in directory pub/tex/sugi-tex-examples. The fontscale file used in the examples is also available from DECUS, as is the source to this paper (which, incidentally, was set entirely in \TeX).

TRADEMARKS

All trademarks and registered trademarks are the property of their respective owners. In particular, SAS is a registered trademark of SAS Institute Inc., Cary, North Carolina. \TeX is a trademark of the American Mathematical Society. METAFONT is a trademark of Addison-Wesley Publishing Company. NeXT is a trademark of NeXT Computer, Inc. PostScript is a registered trademark of Adobe Systems Incorporated. Word is a registered trademark of Microsoft Corporation. Word Perfect is a trademark of Word Perfect Corporation. X Window System is a trademark of MIT. LaserJet is a trademark of Hewlett-Packard Co. UNIX is a trademark of AT&T. VAX and VMS are trademarks of Digital Equipment Corporation. Macintosh is a trademark of Apple Computer.

CONTACTING THE AUTHOR

Joseph E St Sauver
Associate Professor/Assistant Director,
Academic User Services
Office of University Computing
235 Computing Center
University of Oregon
Eugene, OR 97403

Phone: (503) 346-1720  FAX: (503) 346-4397

BITNET: JOE@OREGON
INTERNET: JOE@OREGON.UOREGON.EDU

789