Abstract

Frequency tables seem simple, yet the SAS® System is unable to produce acceptable tables; problems include: severe limits on the length of variable labels and formats, no ability to store extended text describing variables (like the text of questions), and limited ability to handle multiple response variables, among others. As a result many intensive users of frequencies, like survey research firms, use their own custom-written software. This paper describes aspects of a project undertaken for the National Opinion Research Center (NORC) to produce the NORC Automated Codebook. It uses the concept of a codebook to consider how the characteristics of data and data analysis are changing in the 1990s. Greater ability to describe and document datasets and variables is becoming important for scientific, marketing, business, and government data.

Introduction

Frequency tables are widely used. For data that are inherently categorical, they are the most common type of statistical summary. Containing labels, values, frequencies, and percentages, frequency tables seem so simple that it may be hard to see why they are worth an entire paper. Yet a close look at SAS frequency tables shows a number of general flaws in the ability of the SAS System to present information to users. These flaws affect much more than just frequency tables, they go to the heart of the ability of users to describe and document their data. To fix these problems, I will suggest a number of changes that would help. The issues here are considerably broader than frequency tables. They impact users of many kinds of data, particularly data which they did not personally collect. A number of trends tend to increasingly separate those who collect data from data analysts. As a result, these issues will be come even more important in the years to come.

Current Frequency tables

Table 1 contains an example of the PROC FREQ output we all know well. There are 5 columns; from left to right they are: formats (value labels), frequencies for each category, the percent of observations in each category, cumulative frequencies, and the cumulative percent.

Each column has a descriptive heading. The whole table is headed by the variable label. The output can be modified with several options on the PROC FREQ or TABLES statements, but the adjustments are minor.

What is wrong?

The frequency table in Table 1 has a variety of problems. Some of the worst are:

- The variable name can be no longer than 8 characters. In many cases, for example large datasets, this forces cryptically obscure abbreviations and makes them harder to use.
- The variable label can have no more than 40 characters. This is arbitrary and, although it is a useful reminder, it is inadequate to provide much information about the variable.
- Formats (value labels) print no more than 16 characters. Although version 6.07 SAS can store as many as 200 characters in formats; in tables it only prints 16. This is not sufficient to label more than the simplest categories. Again, it forces cryptic abbreviations and makes work harder.
- There is no automatic way to print both the format and the value simultaneously.
- The entire table occupies 62 columns, no more and no less. Even when you are printing on paper or screens with 100 or 200 columns available, the table still contains only 62 columns surrounded by an expanse of white space.
- There is no way to print weighted and unweighted percentages or frequencies in the same table.
- There is no way to print columns of valid and total percentages (including missing data) in the same table.

<table>
<thead>
<tr>
<th>Q4</th>
<th>Frequency</th>
<th>Percent</th>
<th>Cumulative Frequency</th>
<th>Cumulative Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEGITIMATE SKIP</td>
<td>199</td>
<td>74.8</td>
<td>199</td>
<td>74.8</td>
</tr>
<tr>
<td>DON'T KNOW</td>
<td>2</td>
<td>0.8</td>
<td>201</td>
<td>75.6</td>
</tr>
<tr>
<td>NO RESPONSE</td>
<td>2</td>
<td>0.8</td>
<td>203</td>
<td>76.3</td>
</tr>
<tr>
<td>CTA bus</td>
<td>39</td>
<td>14.7</td>
<td>242</td>
<td>91.0</td>
</tr>
<tr>
<td>Illinois Central</td>
<td>17</td>
<td>6.4</td>
<td>259</td>
<td>97.4</td>
</tr>
<tr>
<td>CTA Elevated tra</td>
<td>2</td>
<td>0.8</td>
<td>261</td>
<td>98.1</td>
</tr>
<tr>
<td>Cab</td>
<td>1</td>
<td>0.4</td>
<td>262</td>
<td>98.5</td>
</tr>
<tr>
<td>Other</td>
<td>4</td>
<td>1.5</td>
<td>266</td>
<td>100.0</td>
</tr>
</tbody>
</table>
Missing data are poorly handled. Either they are not included in the table or they sort to the top. Printing them in the first lines tends to exaggerate their importance and is not typically where users want to see them.

Multiple response data can be handled only awkwardly, with complicated work-arounds. Even in PROC TABULATE, it’s difficult to handle multiple response data.

Frequency tables are available in only this one, inadequate format. Other formats, available in the REPORT procedure, require custom programming.

Only two output formats are available: a single frequency table can be written to a SAS data set in a single pass or tables can be output, in the same format they are printed (ASCII or EBCDIC). This forces many people to spend time retying data from SAS tables into tables and reports.

The conclusion that I reach from this: Standard SAS frequency tables are minimally adequate. I am not alone in this conclusion. These and other problems have forced many organizations which make intensive use of frequency tables to write their own frequency table programs.

The NORG Automated Codebook

The National Opinion Research Center (NORG) conducts public opinion surveys for government, business, and academic clients. The inadequacies of SAS frequencies forced it to write its own program, called the NORG Automated Codebook. A codebook is a frequency table with special characteristics. Codebooks were developed because public opinion researchers discovered that respondents tended to answer differently depending on the exact text of the questions and responses and the order of the questions and responses. Therefore, survey data analysts require not just abbreviated summaries, but the complete text of all the information given to the respondents and the full text of all possible responses in the exact order in which they were presented. This full-text information, when combined in tables with frequencies and percentages, is called a codebook. Thus a codebook is a kind of enhanced frequency table. The automated codebook is written using the SAS System and runs on DOS, Windows, and Unix.

The goal of the Automated Codebook is to produce the complete text of a codebook without additional work on the part of users. Questions and responses are in machine-readable form in a computer-aided telephone interviewing (CATI) system. The CATI system writes the data to an ASCII file. Rather than retyping all this information, the NORG Automated Codebook reads it directly and transforms it into SAS statements. It constructs SAS files from the data, complete with variable labels and formats. Most importantly, it prints out a codebook. An example fragment from a codebook is in Table 2. Several characteristics make this more useful than the standard SAS frequencies output. First, it is considerably more compact. Second, the codebook contains not only variables and labels, but the full text of questions and possible responses. It also includes both total and valid percentages in the same table. Thus it is considerably more informative.

A codebook has a single goal: to show the complete context in which the data were collected; to show exactly what the subjects were responding to when they were asked the questions.

What would be better?

The NORG Automated Codebook is only the beginning and I present it here more to illustrate the possibilities than for any other reason. It is being enhanced to provide additional capabilities. Much more can be done to improve the information users receive about their data. Here are some of my suggestions:

Longer formats. Some limit on formats may be necessary, but the current printing limit of 16 characters in a frequency table is hopelessly out of touch. If the linesize option is greater than 62 characters, longer formats should print. If the format is longer than the linesize, an option on the TABLES or FREQ statement should allow lines to wrap. See Table 2, question 4 for an example of how this looks.

Variable domains. Many variables can only take on values within a restricted range. Gender, for example, usually can have two values plus missing data. For each variable, the pool of valid values is called the domain of a variable and it should be stored along with the name in the data dictionary. By default the domain would be all floating point numbers. Special domains include all positive real numbers, all integers, and all positive integers. Domains can also specify the desired order in which to print the values. In addition to making it easier for users to validate data (functions could be provided to compare input data with the domain), domains would solve the problem of printing values with a frequency of zero. Frequency tables and crosstabs are designed as data-driven, they should be domain-based. The structure of the output for frequency tables or crosstabs should be the same regardless of the number or distribution of responses because the structure should be based on the domain not the data.

Multiple variable names: SAS requires variables to have one and only one name, but people refer to variables by many names; for example, by position as in position in a questionnaire, e.g. Q127, or by content, e.g. AGE2 or A_MOTH. One name would be the default name to be printed when a range of variables is specified (e.g. _numeric_).

Variable notes. We often need to store additional textual information that cannot be kept in a variable label; I call this text a variable note. The difference between a variable label and a variable note is that the label contains an abbreviated description of the variable. The variable note contains unabbreviated information: the full text of questions; annotations and notes made when the data were collected; decisions about coding, missing data, and correcting data errors. In short, a variable note provides a place to store metadata; that is, information about data.

Assignment statements and transformations used to construct or edit the variable should be stored automatically along with the time and date. If frequencies and descriptive statistics were stored in a variable note, users would have a simple way to check data integrity. Many datasets are continuously updated and variable notes would help users manage the problems that this causes.

Functions need to be available to edit, view, search, format, and parse the text in the notes, including converting text into a variable and vice versa. Upon user request, notes would print at the bottom of statistical output. They could also be printed as an option on the CONTENTS procedure. Since variable notes do not have the same variable-and-observation structure as ordinary data, modern programming techniques allow them to be of unlimited length.
File notes. In addition to information about variables, users need to store information about an entire file; I call this text a **file note**. File notes describe an entire data set: who collected it, who funded it, why and how the data were collected, the conditions under which it was collected, how the sample was constructed, details of the experimental protocol, etc. File notes should contain the name of the prior data set(s) and any subsampling information automatically. Like variable notes, facilities should be available to search and manipulate file notes. The length of a file note should be unlimited.

**Pictures, Video, and Audio.** Personal computers greatly enhanced the number of people who have access to computerized video, audio, and graphical displays. These media are increasingly important in research. Focus group tapes and summary tapes are a research standard. Non-textual information like cards with pictures is increasingly a part of survey research. As this information becomes part of the research process, we need ways to integrate it more closely with the data we use for statistical analysis. For the same reasons that we need variable notes and file notes, we need to be able to store many kinds of media in our data files.

In addition to issues of what is stored in a data set, there are also issues related to how the data are displayed. One of the SAS System's great weaknesses is its rigid, limited display formats. Some of the possibilities for new displays include:

**Multiple print formats.** Frequency tables have a simple structure and only about 10-12 columns of information (labels, values, frequencies, cumulative frequencies, percents, cumulative percents, percents with and without missing data, various weighted frequencies and percents). Options can easily be implemented to allow users to specify which columns they want to print.

**Multiple output formats.** The world consists of much more than just ASCII or EBCDIC files, and SAS data sets. Because SAS tables support no other formats, many users spend significant amounts of time retyping data into more usable reports. This is a waste of time. Users need the ability to write out tables in many file formats: minimally, as spreadsheet files (.WK1), tab delimited files, or comma-delimitied files. In addition, it would be very useful to be able to save more than one table into a SAS file in a single pass of the data.

**Interactive codebooks.** Although batch programs are a powerful tool, interactive work is where many of the possibilities in this paper come together. Most interactive programs do not take advantage of the flexibility of a computer display. Users should be offered:
● A choice of several output formats. Menus should allow specification of exactly what text, graphics, and statistics are displayed. Why can't users see the video?

● A choice of numerical output options, including numbers, percents, bar charts, histograms, stem-and-leaf plots, box plots or other appropriate displays. For continuous variables, cutpoints for bins should be interactively definable by users.

● Keyword searches on the text of variable names, labels, notes, and formats. This is particularly valuable for large or complex datasets where information may be scattered among thousands of cryptically named variables.

● Annotations. Users should have the ability to annotate questions as they view them and to show or hide previous annotations. A browse mode should allow viewing, but no annotations.

● Flexible output. Users should be able to print or send to a file, anything they can see on the screen, in any format.

This short list does not exhaust the possibilities for more flexible, more complete display of information, but it indicates some valuable possibilities. The real problem that these suggestions point to is that obtaining simple summary information about a SAS data set and its contents is requires too much work. I believe that it is possible to design single one-line command or dialog box that would allow users to define output for any summary statistics. Simply and metaphorically: Users should be able to push one button and receive a summary of any variable in the format of their choice.

Who would benefit?

In recent years we have seen much attention paid to the strategic value of information. This is apparent not only in news stories, but also in the development and increasing importance of organizational structures like information systems divisions in corporations, the appointment of chief information officers, vice presidents of information systems, and the development of executive information systems. For information to be used strategically, requires extensive data collection and analysis. As information spreads more widely, the people who collect it are increasingly separated from the analysts and consumers of the information. This means that secondary data analysis is becoming more common and more important. This may be relatively new in many settings, but one area has relied on secondary data analysis for over a half century; public opinion survey analysis.

We can learn a great deal by looking at the techniques that survey analysts have developed to help them analyze data they didn't collect. If we pay attention to their half-century of experience, one of the first things that draws attention is that the people who collect survey data routinely provide much more detailed, sophisticated, complete descriptions of their data than is common elsewhere. The concept of a codebook is one major illustration of their sophistication. The descriptive, textual, and numeric information in a codebook provide researchers with a clear sense of what was happening when the data were collected.

The problem is that the SAS System does not contain facilities to allow SAS data sets to be very well described at all. To cope with this problem secondary analysts rely on paper documentation including paper codebooks to describe the data and the collection procedures. There are serious weaknesses with this approach. The paper is invariably out of data and incomplete. The paper is easily separated from the data. Keeping the data and the paper filed so that they are both accessible is an expensive, time-consuming task. Many data libraries have special collections of data sets that no one knows what they contain. During periodic housecleanings, these are thrown away. The paper is much harder to use than a good interactive system which relied on machine-readable information stored in the data set.

Data are collected in many ways: experimental protocols in agriculture in agriculture, psychology, or medicine; forms filled out by clerks, and automated data collection devices among others. Data analysts would benefit from being able to read, hear, or see what the details of the data collection protocol, or how the variables were constructed or transformed. Underlying this is a universal need, a need for more information about the data than is available from standard SAS. If SAS Institute made it easier to store information about the data, and made it easier to display this information in a variety of useful ways, almost all researchers would benefit.

References


Acknowledgments

A more complete treatment of these issues will be published in Blank, G. (1993), "Codebooks in the 1990s or Aren't You Embarrassed to be Running a Multimedia-Capable, Graphical Environment Like Windows and Still be Limited to 40-Byte Variable Labels?" Social Science Computer Review, 11. I would like to thank Howard Speizer who directs the NORC Automated Codebook project and Valerie Cooke of NORC who worked on it with me.

SAS is a registered trademark of SAS Institute Inc. in the USA and other countries. © indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

Contact

Grant Blank
1700 East 58th Street #2307
Chicago IL 60637-1935
312/947-8194
grant@sam.spc.uchicago.edu

Notes

1. You can work around this by typing the value as part of the format.

2. This is also true of PROC TABULATE output.

3. The SAS System is not alone in its inadequacies. Alternative statistical and database programs have similar or worse weaknesses.

4. Variable domains are related to the relational database concept of a domain (Date 1985, p. 236-239).

5. To repeat, PROC REPORT partially compensates for this, but at a cost of requiring custom programming in an old-fashioned, not-user-friendly environment.

6. See David and Robbin (1990) for one attempt to provide users with easy access to information about data.