The Stata Journal (2004)
4, Number 2, pp. 190–215

Speaking Stata: Graphing categorical and compositional data

Nicholas J. Cox
University of Durham, UK
n.j.cox@durham.ac.uk

Abstract. A variety of graphs have been devised for categorical and compositional data, ranging from widely familiar to more unusual displays. Both official Stata commands and user-written programs are available. After a stacking trick for binary responses is explained, bar charts and related displays for cross-tabulations are discussed in detail. Tips and tricks are introduced for plotting cumulative distributions of graded (ordinal) data. Triangular plots are explained for three-way compositions, such as three proportions or percentages.

Keywords: gr0004, graphics, categorical data, binary data, nominal data, ordinal data, grades, compositional data, cross-tabulations, bar charts, cumulative distributions, logit scale, catplot, tabplot, tableplot, distplot, mylabels, triplot

1 Introduction

Given the new graphics introduced in Stata 8, *Speaking Stata* has turned to discuss graphics directly. In the previous column (Cox 2004), we started with the fundamental issue of graphing univariate distributions. We now focus on graphing categorical and compositional data, with particular emphasis on ways of going beyond what is obviously and readily available in official Stata.

William Cleveland, the author of two key graphics texts (Cleveland 1993, 1994), has suggested that "for 80% of all datasets, 95% of the information can be seen in a good graph" (Bentley 1988, 60). As far as many categorical datasets are concerned, one might wonder if they fell in the other 20% or if that good graph were proving extraordinarily elusive. Indeed, graphics and categorical data are not obvious bedfellows. A common caricature runs that, with measured data, you should start with simple graphs, while with categorical data, you should start with simple tables. But like even good caricatures, this picture is both true and false and needs qualification.

First, simple bar and pie charts of categorical data will have been familiar to many readers since childhood, but they are not covered by many introductory texts. Perhaps they are considered too elementary or too trivial. One good exception is Wild and Seber (2000). At another extreme, many categorical data analysis texts in the 1970s and 1980s stressed the use of log-linear and logistic models but paid little or no attention to graphical representation. On the other hand, several recent texts (e.g., Lloyd 1999; Agresti 2002; Simonoff 2003) are more concerned with emphasizing the strong links between categorical data analysis and other branches of statistics. For that and other reasons, such texts pay greater attention to graphics. In addition, and going beyond what can

be considered here, there is a substantial but fairly self-contained literature on biplots and related displays, many of which are designed for categorical data (Gower and Hand 1996; Blasius and Greenacre 1998; Friendly 2000). (Ulrich Kohler has a Stata program called `biplot` on SSC.)

In this column, we will start with a simple stacking trick for binary responses. Then we will look in more detail at bar charts and variations on them. Simple they may be, but in many ways they remain among the most effective graphics for categorical data. Then we will examine some displays that perform well for ordinal data, specifically graded data. Finally, we will look at a standard triangular plot for compositional data, specifically for the case of three categories whose percentages necessarily sum to 100, allowing a two-dimensional representation.

There are two key references throughout this column. The first is the *Stata Graphics Reference Manual* [G]; the second is the most useful and complementary compendium of examples given by Mitchell (2004).

2 Kinds of categorical data

Let us first review some terminology: depending on your field, several terms are likely to appear standard, but a few are nonstandard. Many texts classify kinds of data, but those classifications seem to coincide only when one text cites another. However, several classifications mix together cross-combinations of (1) discrete versus continuous (a standard mathematical distinction); (2) nominal, ordinal, interval, and ratio (a scheme introduced by Stevens 1946); and (3) fuzzier distinctions, such as categorical versus measured and qualitative versus quantitative (authors differ on where to draw the line).

Categorical data are regarded here as those for which the original raw data are qualitatively distinct categories, rather than quantitative measurements. Those raw data may be quickly converted into quantitative codes, counts (frequencies) over categories, or proportions or percentages of occurrence of categories. They may commonly be received by users in some such converted form.

Binary (Boolean, dichotomous, quantal) data take only two states, which are often coded 0 and 1. They will be very familiar to almost all readers, common examples being yes or no, alive or dead, success or failure, and so forth. Polytomous or multistate data (often loosely called nominal) may take on three or more states, but those states lack a natural order. In practice, many of the often-quoted examples are not clear-cut, partly because of the temptations of giving any scale an ordered interpretation. Thus, marital status (single, married, divorced), occupations, colors, diseases, and political parties can often be ordered, at least approximately, from some viewpoint.

Ordinal data may take on several states, but those states do have a natural (meaning uncontroversial) order. In practice, ordinal data span an enormous range. At one extreme, an ordinal variable may take on a relatively small number of ordered categories (sometimes called 'grades'), such as opinions on some defined scale from 'Strongly agree' to 'Strongly disagree'. Then there are ranks, which are, in the strict sense, permutations

of integers 1 up and so possess much structure (Diaconis 1988; Marden 1995). From another viewpoint, ranks are akin to counts. We must also consider test scores or other composite scales (e.g., assessments of disability or pain) that are often treated as if they were continuous measurements to some approximation. (Where would educational systems be without grade-point averages or the equivalent?)

Graphically, these differing kinds of categorical data present unequal challenges. Binary data often invite—indeed demand—a reduction to probabilities of one or other possible outcome, which can then be plotted directly. The main difficulty may then lie in showing data points that are all either 0 or 1, or more generally, just one of two possible values. However, one simple and perhaps unfamiliar technique is presented here to begin with. At the other extreme, the closer ordinal data are to quantitative scales that are nearly continuous, the less they need any special form of graphical treatment. The main needs for distinctive graphical forms therefore arise with polytomous or coarse ordinal scales.

3 A stacking trick for binary responses

Suppose that we have a binary response (coded 0 and 1 in most examples) and that an important covariate also shows some discreteness, if only as a consequence of the resolution of recording. In the `auto` dataset,

```
. sysuse auto
```

one such problem is illustrated by figure 1:

```
. scatter foreign mpg, ylabel(0 1, valuelabel angle(h))
```

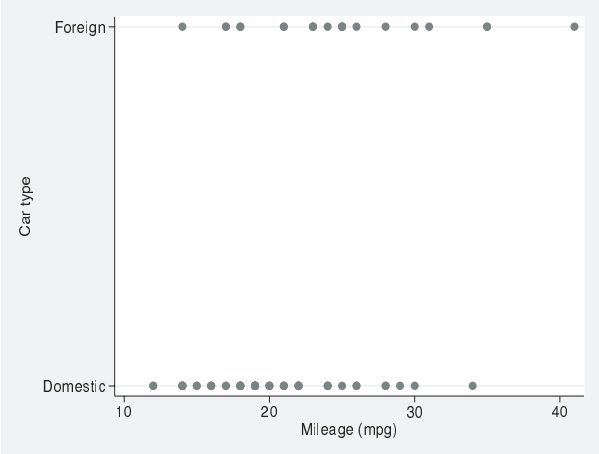

Figure 1: The relationship between car type and mileage is obscured by overplotting of identical data points.

This graph shows many fewer plotted points (30) than there are observations (74) because of ties on both variables, but we have no way of reading any frequencies off the graph. Such ties will often occur with many kinds of data, as with, say, patients' ages in years or incomes recorded in rounded terms. You may already know a solution to the graphical problem: use the `jitter()` option. Jittering does not always work very well, especially if you want a graph for public use. It may also require more explanation in print (or more argument with reviewers) than the trick to be explained now, which is a `dotplot`-like idea of little bars of points stacked vertically. Here is one recipe. As with all demonstration cookery, the details have been customized slightly so that the example works well, but the main idea is easy to grasp.

What is the largest number of ties? We can read that from the results of `tabulate mpg foreign` or calculate it directly by

```
. bysort foreign mpg : gen freq = _N
. summarize freq
```

which in our example shows a maximum of 8. (This last technique may require more care in the presence of missing values, not an issue here.) Let us decide that we want the longest bars to be of length 0.1. We can now produce a variable to be plotted in a single command:

```
. bysort foreign mpg : gen foreign2 =
> cond(foreign == 1, 1 - 0.1 * (_n - 1)/7, foreign + 0.1 * (_n - 1)/7)
. label val foreign2 origin
```

However, that single command does deserve some explanation. The `cond()` function specifies what is to be done both when `foreign == 1` is true and when that is false. The `foreign +` is clearly superfluous when `foreign` is 0, which is the only other value occurring in the `auto` data. For a more general solution, we should worry about dealing properly with missing values, which will map to missing whenever anything is added.

Within each group of tied values on `foreign` and `mpg`, the built-in variable `_n` varies from 1 to 8 (as just ascertained), so `(_n - 1)/(_N - 1)` in general and `(_n - 1)/7` in our example vary from 0 to 1. Thus, in constructing a variable to be plotted, we subtract (at most) 0.1 from 1 and add (at most) 0.1 to 0. The details to tune according to taste and circumstance are thus (a) the constant, which is 0.1; (b) the maximum frequency -1; and (c) whether bars are added to or subtracted from the horizontal reference lines for 0 and 1.

```
. scatter foreign2 mpg, ylabel(0 1, valuelabel ang(h))
> ytitle(`"`: variable label foreign'"')
```

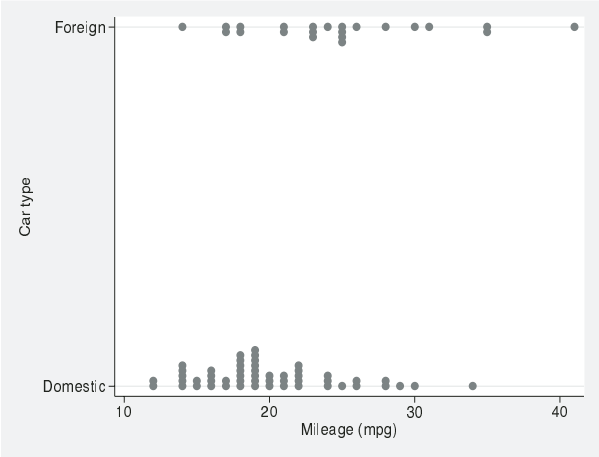

Figure 2: Data points are piled in little stacks to show frequencies of combinations.

Figure 2 shows the result. Note a small flourish here, automating the use of the variable label. As it happens, with this dataset just typing it in would take no more effort, but a general method would be preferable in a do-file or program that you might write for wider application.

4 Bar charts of frequencies

One of the simplest kinds of plots, but one often requested, is a bar chart of frequencies of one or more categorical variables. If we consider what is available in official Stata, the natural commands to consider here are `histogram` and `graph bar` or `graph hbar`.

4.1 Using histogram

`histogram` is optimized for the case of continuous variables; by default, you get a series of touching bins. You can spell out that you want a graph emphasizing a discrete scale by adding the `discrete` option. You can specify gaps between bars, which are customary with categorical scales to show that the scale really is *not* continuous, by using an option such as `gap(50)`. You can insist that value labels, likely to be attached to most categorical variables, be shown by using another option, `xlabel(, valuelabel)`. These tweaks may be sufficient for some variables, but in total, `histogram` may remain problematic for categorical scales.

First, several lengthy value labels may be difficult to read when placed side by side along a horizontal axis. Changing the font size or the orientation of the text may just replace one problem with another: at one extreme, we approach giraffe graphics in which readers are presumed able and willing to move their heads over a range of angles to scrutinize the plot presented. However, using the `horizontal` option could be sufficient to solve this.

Second, many designers of bar charts and perhaps even more readers prefer a display more colorful than the monochrome display standard with histograms.

Finally, `histogram` can be extended to a two-variable classification, as a multiple panel display is obtainable with the `by()` option. Sometimes this works well, but equally sometimes you would prefer something different.

4.2 Using graph bar or graph hbar

`graph bar`, `graph hbar`, and, for that matter, `graph dot` offer a different approach. It is easiest if the frequencies come predefined as a variable because each command can then be used with either `(asis)` or `(sum)`. But if you want Stata to do the counting for you, you must do things another way. In particular, with the `auto` data read in,

```
. graph hbar (count) rep78
```

does not give you the frequencies of the categories of `rep78`. It counts nonmissing values of `rep78` and shows a single bar of height 69. Moreover,

```
. graph hbar (count) rep78, by(rep78)
```

is illegal. More positively, something like

```
. graph hbar (count) mpg, by(rep78)
```

does what you want but at the cost of a mislabeled graph and some strain on the brain. Incidentally, being able to say why this works as a bar chart of the frequencies of `rep78`, apart from the mislabeling, is a little test of your understanding of what is going on.

The underlying problem here is that `graph hbar` and its siblings `graph bar` and `graph dot` are built around a temporary `collapse` of the data, whereas in effect what we want here is a temporary `contract`. However, that problem is a little difficult to diagnose without looking inside the graphics code.

Another way to do it is by calculating something in advance, as in

```
. generate freq = 1
. graph hbar (count) freq, over(rep78)
```

and yet another way to do it is to `contract`, as in

```
. contract rep78
. graph hbar (asis) _freq, over(rep78)
```

The first of these may smack of a programmer's trick, while the second has the strong disadvantage that the original dataset will have to be read in repeatedly to do a series of bar charts or even to return to the original data for further analyses.

Arguably, none of these solutions is good, especially when we consider the common need for also showing fractions or percentages. In addition, we would expect a bar chart command to handle smoothly any missing values or `if` or `in` conditions.

4.3 Using catplot

`catplot` is a convenience command designed to avoid these awkwardnesses. The aim is simply that a bar or dot chart of frequencies be available through a single command, without any need for preparation or restructure of the dataset. `cat` is meant to suggest category, naturally, but any feline undertones should be regarded as felicitous. You may install `catplot` from SSC using the `ssc` command ([R] **ssc**):

```
. ssc install catplot
```

To be precise, `catplot` shows frequencies (or, optionally, fractions or percentages) of the categories of one, two, or three categorical variables. The first named variable is innermost on the display so that its categories vary fastest along the axis. The syntaxes `catplot bar`, `catplot hbar`, and `catplot dot` indicate use of `graph bar`, `graph hbar`, and `graph dot`, respectively. The choice is a matter of personal taste, although in general horizontal displays make it easier to identify names or labels of categories and so avoid the giraffe graphics just deprecated.

The default display with `bar` and `hbar` is graphically conservative, reflecting the view that height of bars and text indicating categories are the best ways of conveying information. If you wish also to have bars in different colors, specify the option `asyvars`, which differentiates the categories of the *first* named variable; to stack bars of different colors, specify the further option `stack`.

The default display with `dot` is similarly conservative. If you wish to have point symbols in different colors, similarly specify the option `asyvars`, which differentiates the categories of the *first* named variable; to use different point symbols, use the further option `marker()`.

There are various simple options to show percentages or fractions. `percent` indicates that all frequencies should be shown as percentages (with sum 100) of the total frequency of all values being represented in the graph, while `percent(`*varlist*`)` indicates that all frequencies should be shown as percentages (with sum 100) of the total frequency for each distinct category defined by the combinations of *varlist*. There are similar fraction options for fractions with sum 1.

The `sort` option specifies that values shown should be sorted in each category (higher values at the bottom of each category). Sorting is applied to all variables shown. The `descending` option specifies that sorted values should be shown in descending order (higher values at the top of each category).

Simple examples with just one categorical variable are

```
. catplot hbar rep78, sort
. catplot hbar rep78, sort desc
```

and with two such variables are

```
. catplot hbar rep78 foreign
. catplot bar rep78, ylabel(, angle(h)) percent(foreign)
> by(foreign, note("") subtitle(Repair record 1978, pos(6)))
```

```
. catplot hbar rep78 foreign, percent(foreign) blabel(bar, position(outside)
> format(%3.1f)) ylabel(none) yscale(r(0,60)) ytitle("")
> subtitle(Repair record 1978 distribution by car origin)
```

The last two of these are shown as figure 3. Almost all the flexibility comes from the `graph` commands for which `catplot` is a wrapper.

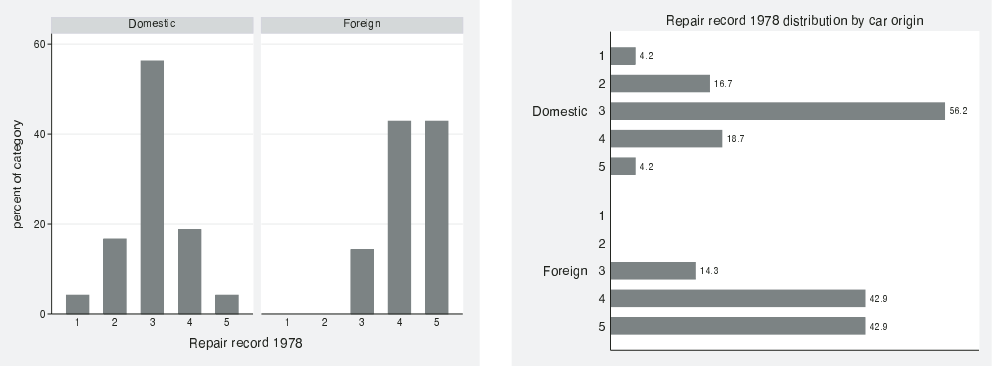

Figure 3: `catplot` can produce bar charts of percentages.

4.4 Hybridizing graphs and tables

A step away from the displays afforded by `catplot` are what may be called graphical tables. Typically, the value in each cell of such displays is represented by a bar, a spike, an interval, or some other similar way. The commands `tabplot` and `tableplot` from SSC provide some flexibility here. Both are applicable to problems with two categorical variables and some third quantity to be plotted for each cross-combination of categories.

`tabplot` is specialized to show counts (or optionally percentages or fractions of total, rows, or columns) and to show vertical or horizontal bars. For the most part, it can be considered as a wrapper program for `twoway rbar`. The name is intended to remind you of `tabulate`, which many users happily abbreviate `tab`, a command centered on showing tables of counts (or percentages). `tabplot` works best in practice when at least one variable is ordinal so that intelligible patterns might be expected in relation to at least one axis. By tuning the `barwidth()` option, the bars can be made very thin, which is not especially useful, or they can be made so wide that they touch, thus giving a set of juxtaposed histograms, which can be very useful. `tabplot` is in fact not restricted to categorical variables; the variable on either axis or the variables on both can be flagged as to be treated literally. Hence, `tabplot` has a secondary use for juxtaposing histograms, but that is not explored fully here.

`tabplot` is primarily intended to show the main structure of a table. It can be useful for exploratory inspection or even public presentation of moderate-sized tables, say, those with about 10 or 20 rows or columns. At that size, even experienced analysts can sometimes find it difficult to see the table for the cells. On the other hand, `tabplot`

gives up on showing detailed numeric scales on either axis. There is just one token concession to showing numbers as such—a note indicates the largest value shown in any cell. Beyond that, `tabplot` relies on users' mental assessment of bar or spike heights or lengths. Fortunately, this is the easiest graphical skill (Cleveland 1994) so that fine as well as coarse structure can usually be discerned. For table look-up of individual cell magnitudes, there is, unsurprisingly, nothing superior to table look-up. More simply put, the graphs produced by `tabplot` and the equivalent tables complement each other. In practice, many thesis advisors and journal editors are reluctant to approve publishing both on the grounds that they show the same information.

An example is provided by a classic dataset (Tocher 1908) on children's eye and hair color in Caithness in northern Scotland, which has been analyzed in various ways in the statistical literature (e.g., Fisher 1940). Historically Caithness has experienced influxes of migrants not only from other parts of Britain but also from other areas, Scandinavia in particular, so its people show very interesting genetic diversity. Both variables may be regarded as ordinal, at least approximately.

A table comes as usual from

```
. tab eye hair [fw=freq]
```

and a basic frequency plot (figure 4) is then given by

```
. tabplot eye hair [fw=freq], horizontal ytitle(, orient(hor))
```

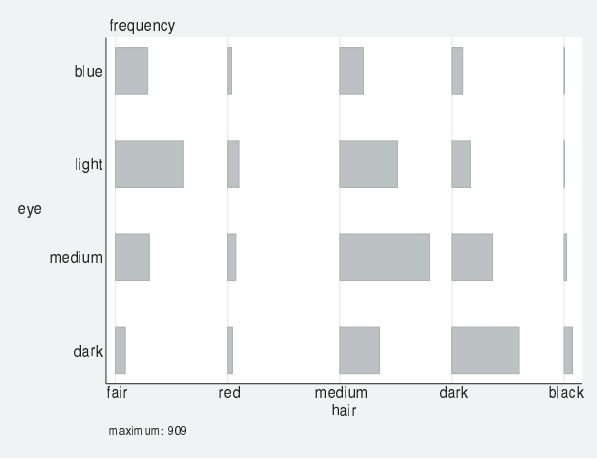

Figure 4: `tabplot` can produce graphical displays of two-way tables of frequencies.

The syntax evidently is designed to resemble that of `tabulate`. In each command, the first-named variable goes on the left of the display (the rows of the table, the vertical axis of the graph) and the second-named variable on the bottom of the display. Note also that, by default, the lowest-numbered row (whenever the row variable is numeric, which is usual but not essential) is shown at the top of the vertical axis.

When all values shown (here the frequencies) are zero or positive, the maximum bar height or length is, by default, 0.8. As the interval between successive categories is, by default, 1, these choices normally imply that neighboring rows and columns remain distinct.

One key scaling allowed is to percentages of either row or column categories. In addition, scaling to fractions is possible, but merely cosmetic in affecting only marginal text and the note about maximum value. One example is thus (figure 5):

```
. tabplot eye hair [fw=freq], horizontal percent(eye) ytitle(, orient(hor))
```

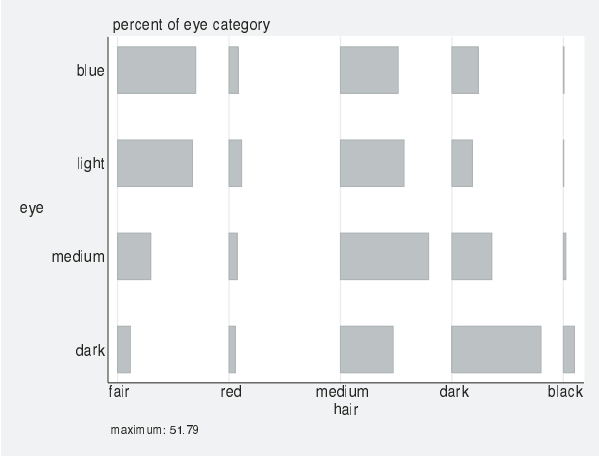

Figure 5: `tabplot` also can produce graphical displays of two-way tables of percentages, here for row categories.

The broad association between eye and hair color should come as no surprise even to rank amateurs in genetics or physical anthropology. There is a diagonal pattern stretching from fair-haired and blue-eyed to black-haired and dark-eyed. Which variable is taken as base for percentages appears arbitrary; experts might be able to advise. In your own field, it will usually be clearer which kind of display is best for your problem and dataset. In this example, we try vertical bars and column percentages (figure 6):

```
. tabplot eye hair [fw=freq], percent(hair) ytitle(, orient(hor))
```

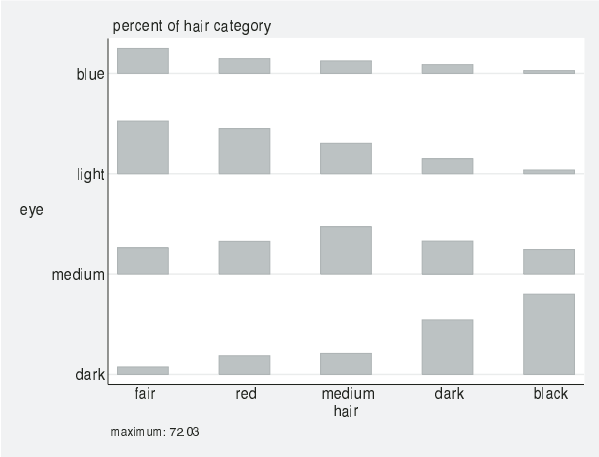

Figure 6: As a variation, `tabplot` here produces vertical bar displays of percentages for column categories.

Let us compare this kind of display with a stacked, or divided, bar chart. Stacking is available through `graph bar` or `graph hbar` or through commands such as `catplot` (figure 7):

```
. catplot hbar eye hair [fw=freq], percent(hair) stack asyvars
> bar(1, bcolor(gs14)) bar(2, bcolor(gs10)) bar(3, bcolor(gs6)) bar(4, bcolor(gs2))
```

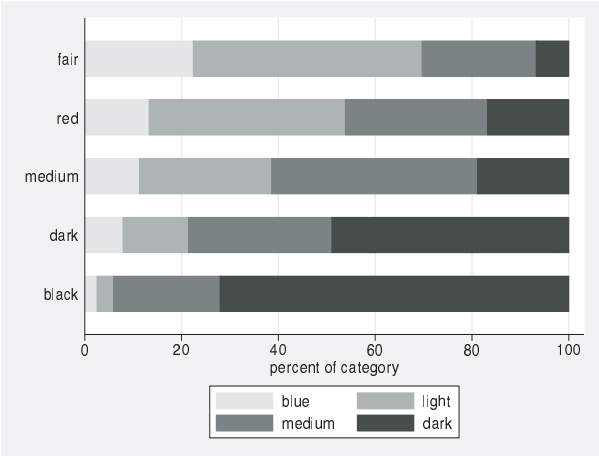

Figure 7: Stacked (divided) bar charts are widely familiar, but only the end bars are anchored to straight base lines, inhibiting the decoding of intermediate category values.

Here, as a small but telling detail, we spell out that bar colors are to be various gray scales (e.g., `gs14`). The main role of color, arguably, is to provide qualitative contrasts.

Nevertheless, it is worth checking that, as far as possible, colors representing ordinal scales do represent a monotonic sequence.

Stacking of percentages, so that divided bars of constant length each represent group totals, can serve as a useful reminder to inexperienced readers of the basis of the graph. Alternatively, it might seem an uninformative tautology: "This graph's main message is that all percentages sum to 100. So, what else is new? I am obliged to look inside each bar for the real information in the data." Naturally much hinges on what you are trying to do and who you have in mind as readers of your graph.

A pattern common with frequencies reduced to percentages is that the percentages in extreme categories behave monotonically, but this is often not true of intermediate categories. Thus from one end of each axis to the other, the percentage of dark or black goes up while the percentage of fair or blue goes down, while other categories sometimes change monotonically and sometimes change through a turning point. All this is no surprise on reflection, but a distinct weakness of the stacked design is that only the bars for extreme categories are based on clear reference lines. It is especially difficult to decode values of intermediate categories accurately; indeed, perhaps few readers try very hard to do this. Clearly, one key advantage of displays like those from `tabplot` is that bar heights, and thus values in the table, may be compared more easily, at least within rows or within columns, because all bars are based on reference lines.

There are other ways of tackling this issue. One is sliding each divided bar so that it straddles a central category. The program `slideplot` on SSC is dedicated to such sliding bar plots, which sometimes work well. Unfortunately, they can also create as many problems as they solve. Nevertheless Stouffer et al. (1949) provided one source with plentiful examples in their monograph, one of the classics of quantitative social science.

`tableplot` is more general than `tabplot`, but to get what you want, you may need to do more work. The main idea, as with `tabplot`, is that two variables provide a framework of rows and columns defining cells. Unlike with `tabplot`, a third variable must be specified. Its values must be unique within each cell defined by the rows and columns. In addition, a range of plot types is available, namely `rbar`, `rcap`, `rcapsym`, and `rspike`. Thus, you can see that `tableplot` is mostly a wrapper for `twoway` with one of the named plot types. In practice, most users seem to refer `rbar`.

The name `tableplot` is intended to be reminiscent of `table`, a fairly general tabulation command, but one for which it is more common to specify precisely what is tabulated. The analogy is not strict and should be promptly forgotten if it does not seem helpful.

Thus the main syntax is schematically `tableplot` *plottype showvar rowvar colvar*. You may also think of this as *plottype z y x*: here y and x define rows and columns, as with `tabplot`, but the third variable, z, gives the heights of the bars, spikes, or intervals plotted within each cell. For example, therefore,

```
. tableplot rbar freq eye hair
```

is equivalent to

```
. tabplot eye hair [fw=freq]
```

Using `tableplot` to do what `tabplot` can do directly would be missing the point, however. More interesting applications go beyond representing tables of counts or percentages. An example dataset comes from a study by Townsend (1995) of various pioneer communities in the Mexican rainforest. In several villages, her team asked how many men and how many women did various tasks, such as doing laundry, cooking, fetching water, fetching wood, and so forth. Their data can be summarized, for each task and each village, on a scale showing distribution of labor by gender:

(number of women − number of men) / (number of women + number of men)

So, if no men do some task, this measure is 1; if no women do it, it is −1; and if equal numbers of men and women do it, it is 0. Depending on your background, you may recognize this as a difference in probabilities, and thus, as similar to several statistical measures or as equivalent to a fractional majority for elections. No doubt other applications exist. Incidentally, note that, in this dataset, the ratio of numbers of men and of women and its reciprocal are both impractical: zeros for some places, some tasks, and both genders would render both ratios indeterminate. Even if that were not true, a serious limit to the usefulness of any sex ratio is its skewness around 1. The scale just defined with its symmetric limits is greatly preferable. The `tableplot` just below required some prior work sorting row and column categories into the best order. Even though neither task nor place is an ordinal variable, sorting rows according to, say, medians across columns and vice versa proved very helpful. More generally, this is often one of the most fruitful steps to identifying underlying structure.

Figure 8 shows the gradation from tasks generally done by women (laundry, cooking, care of hens, fetching water) to those generally done by men (preparing land, harvesting, care of cows, fetching wood). Interestingly, the two middle tasks differ: gardening and milking cows both average about an equal division of labor. In the case of gardening, this is fairly consistent between places, but with milking cows, there is much more variability between places. Subject-matter experts might be able to comment, but the graphical point is simple. This plot represents a 10×12 table and allows both coarse and fine structure to be seen. The decision to put tasks on the rows and places on the columns is clear-cut, given the nature of variability in the data. There is much more variation between tasks than between places, as could be explored further in a formal analysis, if it were thought fit to treat the data as if they were a random sample. Also, the tasks are easy even for nonspecialists to think about, but the place names mean little except to those familiar with the area. Being obliged to put the names vertically does supply an instance of giraffe graphics, but it is the easiest sacrifice to be made. Other solutions, such as abbreviating the names, writing them at 45°, or using a smaller font size, would here gain nothing.

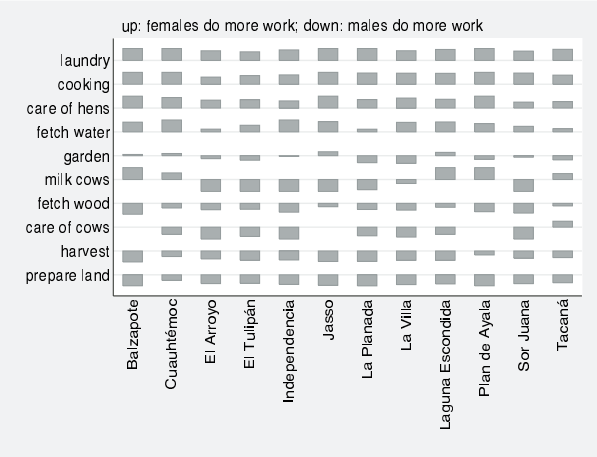

Figure 8: `tableplot` permits the display of values of a third variable by combinations of two variables. Here the division of labor in some Mexican villages between females and males is shown for various tasks.

5 Cumulative distributions for graded variables

A very different approach for ordinal data, especially graded variables, is based on another standard kind of graph, showing cumulative distribution functions. `distplot` (Cox 1999, 2003a,b, 2004) produces plots of cumulative distribution functions or their reverses.

One particular option was introduced with graded data specifically in mind and is especially appropriate for showing data with a relatively small number of categories. `midpoint` specifies the use of midpoints of probability intervals for each distinct value so that the cumulative probability P for a variable X is defined as

$$\Pr(X < x) + \frac{1}{2}\Pr(X = x)$$

With terminology from Tukey (1977, 496–497), this could be called a 'split fraction below'. It is also a 'ridit' as defined by Bross (1958); see also Fleiss, Levin, and Paik (2003, 198–205) or Flora (1988). Yet again, it is the mid-distribution function of Parzen (1993, 3298) and the grade function of Haberman (1996, 240–241).

Using this definition rather than $\Pr(X < x)$ or $\Pr(X \leq x)$ means that more use is made of the information in the data. Either alternative would always mean that some probabilities are identically 0 or 1, which tells us nothing about the data. In addition, there are fewer problems in showing the cumulative distribution on any scale for which the transform of 0 or 1 is not plottable. This approach for graded data was first implemented in Stata by Cox (2001). Its roots go back at least to Tukey (1961), a paper that was, however, not published until 1986.

To develop that point, `distplot` has an option useful for graded data. `trscale()` specifies the use of an alternative transformed scale for cumulatives. Stata syntax should be used with `@` as a placeholder for untransformed values. So, to show probabilities as percentages, specify `trscale(100 * @)`; on an inverse normal scale, specify `trscale(invnorm(@))`; on a logit scale, specify `trscale(logit(@))`; and on a cloglog scale, specify `trscale(cloglog(@))`.

Further information on transformations for probability scales is available in Tukey (1960, 1961, 1977), Atkinson (1985), Cox and Snell (1989), and Emerson (1991). Some of the possible transformations appear as link functions in the literature on generalized linear models (e.g., McCullagh and Nelder 1989; Aitkin et al. 1989).

Let us look at some examples with this little toolkit in mind. For Stata users, repair record `rep78` in the `auto` data is a simple and familiar example of a graded variable. The first step is to look at plain cumulatives (figure 9):

```
. distplot connected rep78, ylabel(, angle(h)) midpoint by(foreign)
```

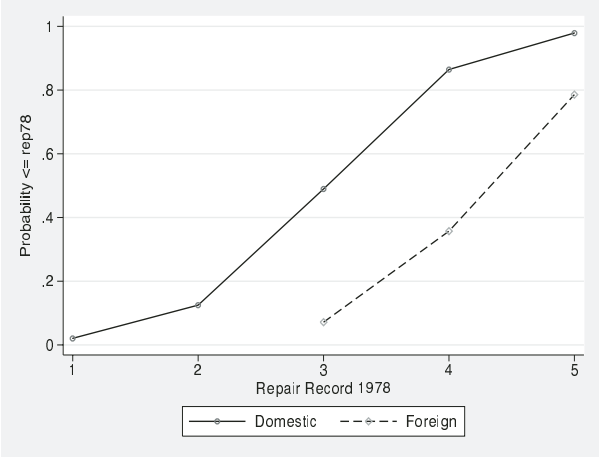

Figure 9: Cumulative distributions of graded variables can be shown using the midpoints of intervals of probability.

The fairly standard S-shape of the cumulative distribution for domestic cars particularly suggests that a transformed scale might yield a complementary view. I like using logits here, for no special reason except that they often work very well, as further examples will show. We can do that on the fly (figure 10):

```
. distplot connected rep78, ylabel(, angle(h) midpoint by(foreign)
> trscale(logit(@)) l2(logit scale)
```

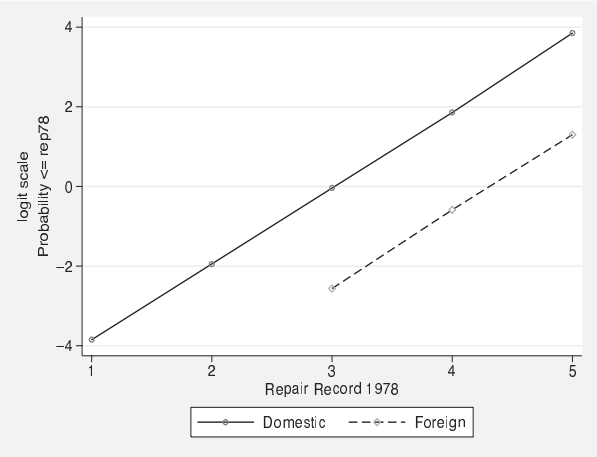

Figure 10: Logits of cumulative distributions of graded variables often plot as nearly straight lines, giving a handle on the difference a covariate makes.

The linearization is dramatic. The first time I got this, I suspected that a silly bug of mine was giving me lines as artifacts. Yet it is genuine. The more general heuristic is simple. First is the empirical observation that logits of cumulatives are often fairly straight. Note that this method is not, in fact, a back-door way of fitting one of the standard logit-based models to ordinal responses. The straightness seems to arise partly from how nature (or society) works and partly from how people devise grading schemes. For example, most schemes are devised so that some occurrences are expected in all categories. Second, do covariates make much of a difference, as shown by very different cumulatives for each group, and if so, is the ordering and even magnitude of effects what would be expected? Third, can extra structure be identified, such as highly anomalous groups or simple interaction effects?

Beyond that heuristic, we need to focus on one tacit assumption that for some will be troubling. Deciding whether cumulatives are straight or some other shape depends on the metric on which graded variables are being shown. The curve shape is contingent, in this example, on taking the values of `rep78` quite literally (meaning, numerically). According to the purists, as represented by Stevens (1946) among others, this is precisely what you should not do with graded data. According to the pragmatists, on the other hand, you should feel free to do whatever works.

In this issue, I tend to side with the pragmatists, despite recognizing the purists' argument. It is always open to users to experiment with different scoring schemes outside of `distplot`. Moreover, if a graph does not help, you just should shrug your shoulders and move on to try other kinds of display. `distplot` with options `midpoint` and `trscale()` has been used to show simple patterns that do not need the gloss of a more formal analysis (Bentley et al., forthcoming). If it helps to select covariates for a modeling exercise, that also can be benefit enough.

Enough arm waving; let us look at some more examples. Fienberg (1980, 54–55) reported data from Duncan, Schuman, and Duncan (1973) from 1959 and 1971 surveys of a large U.S. city asking, "Are the radio and TV networks doing a good job, just a fair job, or a poor job?" Suppose that, underneath the labels below, `opinion` runs 1/3. `group` here evidently is a cross-combination of year and race, created off stage by the `egen` function `group()`. Mapping two or more covariates to one is a standard device here, so long as the total number of cross-combinations remains manageable.

```
          group    opinion      freq
 1. 1959 Black       Good        81
 2. 1959 Black       Fair        23
 3. 1959 Black       Poor         4
 4. 1959 White       Good       325
 5. 1959 White       Fair       253
 6. 1959 White       Poor        54
 7. 1971 Black       Good       224
 8. 1971 Black       Fair       144
 9. 1971 Black       Poor        24
10. 1971 White       Good       600
11. 1971 White       Fair       636
12. 1971 White       Poor       158
```

With these data, we will go straight to logits (figure 11):

```
. distplot connected opinion [w=freq], ylabel(, angle(h)) midpoint by(group)
> trscale(logit(@)) xlabel(1/3, valuelabel) l2(logit scale) legend(col(1)
> position(5) ring(0))
```

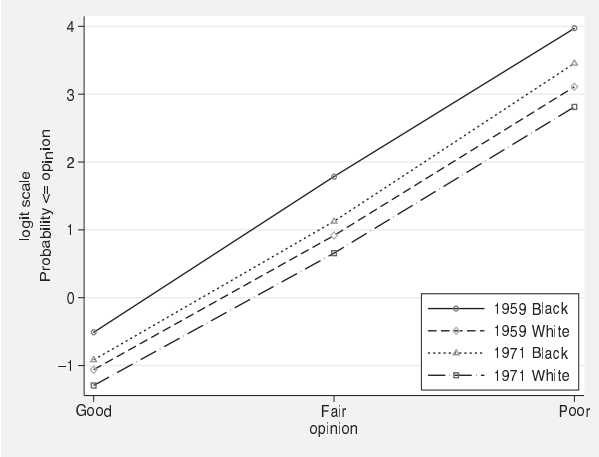

Figure 11: Opinion of radio and TV networks varies with race and has shifted over time, with a narrowing gap between black and white. Higher curves show collectively more favorable opinions.

This shows a clear shift of opinion towards Poor from 1959 to 1971 and a narrowing gap between Black and White. Otherwise said, race and year both make a difference,

which seems no surprise. Whether the narrowing gap is modeled with some kind of interaction effect is an issue for more formal analyses.

As a third and final example, Knoke and Burke (1980, 68) gave data from the 1972 U.S. General Social Survey on church attendance. Suppose that, underneath the labels below, `attend` runs 1/3.

```
                group    attendance      freq
 1. young non-Catholic           low       322
 2. young non-Catholic        medium       122
 3. young non-Catholic          high       141
 4.   old non-Catholic           low       250
 5.   old non-Catholic        medium       152
 6.   old non-Catholic          high       194
 7.     young Catholic           low        88
 8.     young Catholic        medium        45
 9.     young Catholic          high       106
10.       old Catholic           low        28
11.       old Catholic        medium        24
12.       old Catholic          high       119
```

The `reverse` option ensures that higher attendance groups plot farther to the right on the graph. There are clear age and denomination effects and an indication of an interaction between the two (figure 12):

```
. distplot connected attendance [w=freq] , ylabel(, angle(h)) by(group)
> midpoint trscale(logit(@)) legend(column(1) position(1) ring(0))
> xlabel(1/3, valuelabel) l2(logit scale) reverse
```

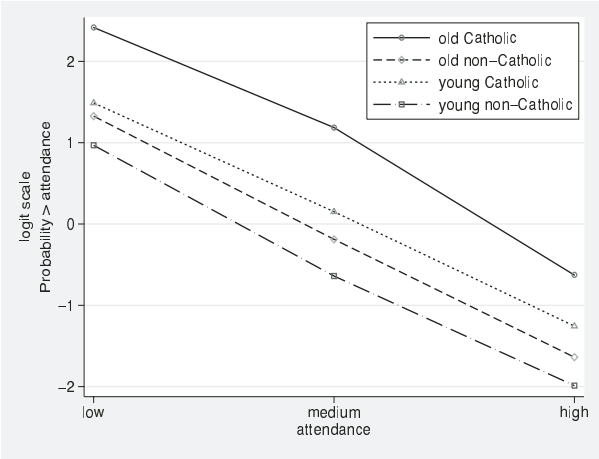

Figure 12: Church attendance is affected by age and denomination. Logits of cumulatives are again nearly straight. On this scale, there appears to be an interaction effect.

6 Better labels on transformed scales

The ability to produce axes on transformed scales may seem a mixed blessing. It raises an issue which is more general. We are happy thinking on a probability scale (about say 0.9, 0.42, 0.007) but might well be less happy thinking on a logit scale (about the equivalents, 2.197, −.323, −4.955 to 3 decimal places). There is a way of getting the best of both worlds. A transformed scale and also labeling in more intelligible terms are possible by using the `ylabel()` and `xlabel()` options to specify text to be shown at axis positions. (By the way, value labels are not general enough to work well, as they can only be attached to integers.)

Suppose that one graph axis is a logit scale, but you wish the axis labels to show untransformed probabilities. Stata could be used as a convenient calculator to work out the mapping, but a dedicated utility is preferable.

The idea behind `mylabels`, which may be downloaded from SSC, is that you feed it the numeric labels to be shown and the transformation being used. It will then place the appropriate specification in a local macro that you name. You may then use that local macro as part of a later graph command. A similar idea may be used for axis ticks: the command is called `myticks` and comes bundled with `mylabels`. The idea behind these programs may be traced to Royston (1996).

The option `myscale()` specifies the transformation. Stata syntax should be used with `@` as placeholder for the original value. Hence, to show original values on a logit scale, specify `myscale(logit(@))`.

The option `local(`*macname*`)` inserts the option specification in local macro *macname* within the calling program's space. If you are unfamiliar with the idea of local macros in Stata, see [U] **21.3 Macros** or Cox (2002). The key idea is, essentially, to put all the definitions together in a bag which can then be referred to concisely. The macro will be accessible after `mylabels` or `myticks` has finished for subsequent use with `graph` or other graphics commands.

For example,

```
. mylabels 0.1(0.1)0.9, myscale(logit(@)) local(myla)
```

means that you have data on a logit scale but wish labels to be displayed that show values from 0.1 in steps of 0.1 to 0.9. What `mylabels` will show is

```
-2.19722 ".1" -1.38629 ".2" -.847298 ".3" -.405465 ".4" 0 ".5" .405465 ".6"
.847298 ".7" 1.38629 ".8" 2.19722 ".9"
```

That is, the text `".1"` will be shown at −2.19722 on whatever axis is specified, the text `".2"` at −1.38629, and so forth. The main point of showing you the list is allowing you to check that you have what you want.

On a graph, you may want plain ticks in between labels. `myticks` creates the list for you.

```
. myticks 0.15(0.1)0.85, myscale(logit(@)) local(myti)
```

Then you can re-issue your graph call (figure 13):

```
. distplot connected attendance [w=freq], ylabel(‘myla’, ang(h)) by(group)
> midpoint trscale(logit(@)) legend(col(1) position(1) ring(0))
> xlabel(1/3, valuelabel) l2(logit scale) reverse
```

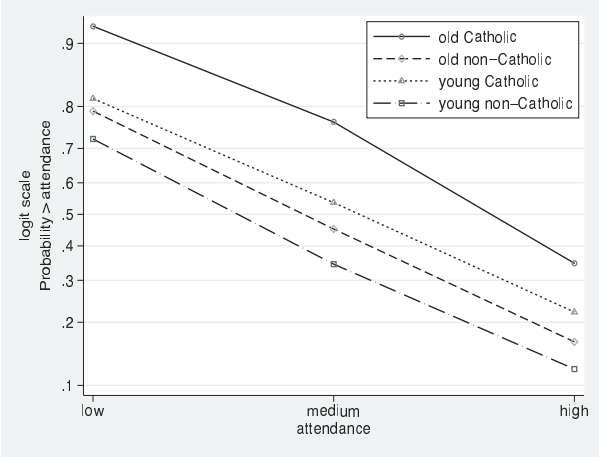

Figure 13: `mylabels` has been used to produce more intelligible labeling of the logit scale in terms of probabilities.

7 Triangular plots for three-way compositional data

Suppose that we have two variables that have a constant sum, such as p = proportion female and q = proportion male, so that $p + q = 1$. A plot of data for those variables shows all points lying on the line defined by that constraint. In practice, we would not draw that plot, unless by accident. We would recognize a bivariate situation as essentially univariate and examine the distribution of, for example, p.

More interesting is the case of three (zero or positive) variables with a constant sum, say three proportions with sum $p+q+r = 1$. This constraint defines a plane in (p, q, r) space. In fact, data points are confined to a triangular subset of that plane, which can thus be laid flat in two dimensions with no loss of information. The trivariate situation is essentially bivariate. Naturally this is just a special case of the more general situation of compositional data (Aitchison 1986), but it is nevertheless an interesting and frequent special case.

`triplot` from SSC produces a triangular plot of three variables, which are plotted on the left, right, and bottom sides of an equilateral triangle. Each should have values between 0 and some maximum value (default 1), and the sum of the three variables should be equal to that maximum (within rounding error). Most commonly, three fractions or proportions add to 1, or three percentages add to 100.

Triangular plots appear under various names in the literature, including percentage or reference triangles and barycentric, mixture, ternary, trilinear, or triaxial plots. Beyond this profusion of terminology there lies a curious pattern of patchy invention and use. Triangular plots have been rediscovered many times over in several fields yet are also apparently little or never used in some other fields, despite the potential for many applications. One root is the barycentric calculus of Möbius (1827); see Gray (1993). Other roots are 18th- and 19th-century studies of color mixing, photoelasticity, and the phase rule (Howarth 1996). Disciplines in which they are popular include genetics (for three genotypes or three alleles), geology (various geochemical compositions and particle shape analysis), pedology (for clay, silt, and sand fractions of a soil; for an early Stata implementation, see Danuso 1991) and political science (election data, often for two big parties and a bundle of others).

A common kind of economic example of triangular plots is based on some three-fold division of activities, say into agriculture (and sometimes other so-called primary activities, such as fishing, forestry, and mining); manufacturing (secondary); and services, including information (tertiary). This classification is some decades old, and some would now prefer to split services and information, producing four sectors, the last occasionally called quaternary. Such a scheme would take the classification beyond the reach of triangular plots. Nevertheless, the trichotomy remains of use for broad-brush description, especially for comparing a range of less developed and developed economies. Some World Bank data on the composition of Gross Domestic Product for 112 economies were used to produce figure 14:

```
. triplot agriculture industry services, separate(Africa) max(100) legend(pos(2)
> ring(0) col(1)) ms(O Th) note(units are %)
```

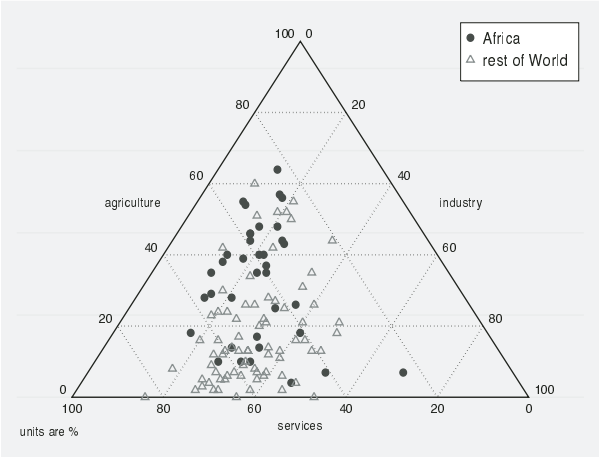

Figure 14: `triplot` is used here to show the structure of Gross Domestic Product in 112 economies in 1996 from World Bank data.

A detail here that may puzzle is the placement of the legend at `pos(2)` and within `ring(0)`. The explanation follows from the kind of Stata graph `triplot` is, underneath its skin. Despite appearances, the triangular plot is just a `twoway` plot with y and x axes removed. The internal grid, the triangular frame, and an optional Y within the triangle are defined by calls to `twoway connect` and `twoway line`, while all else is done by calls to `twoway scatter`. The clumsiest part is the programming of the axis labels, which do not have the flexibility of standard y or x axis labels. The nicest part is that almost all the handles you need are available as standard `twoway` features.

To explain the legend puzzle, the legend is within the plot region defined by the y and x axes. Those axes have been removed, but they nevertheless retain a shadowy existence. Other details of the `triplot` call should be more transparent. The `separate()` option subdivides data points into groups, here according to a binary variable (African or not).

A second example looks at how such compositions change over time. Given data on the composition of the civilian labor force in the United States (Beniger 1986), we can try preliminary plots, not shown here, and then tweak the label positions away from their defaults (figure 15):

```
. generate pos = 3
. replace pos = 9 if inlist(year, 1900, 1920)
. replace pos = 10 if inlist(year, 1870, 1980)
. replace pos = 12 if inlist(year, 1880, 1970)
. triplot ag ind tert , c(l) max(100) mlabel(year) mlabsize(*0.7) clpat(solid)
> mlabvpos(pos)
```

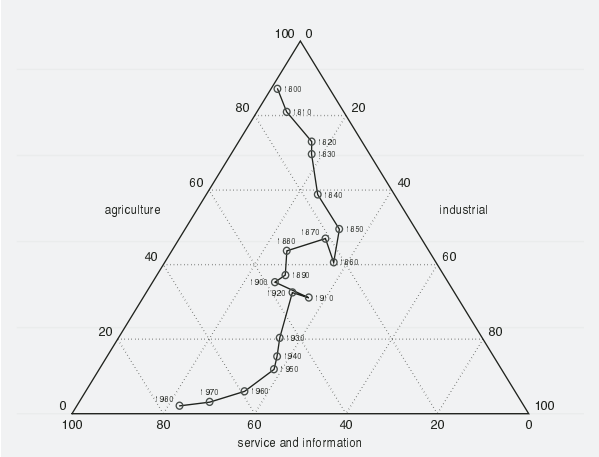

Figure 15: `triplot` is used here to show the changing composition of the U.S. civilian labor force from 1800 to 1980.

Again, we can reach through `triplot`, set up marker labels, and control their size and position, all by virtue of `triplot` being built upon `twoway`. In this case, the trajectory is fairly smooth from the highly agricultural economy of the early 19th century through a more industrial economy to nearer the present, dominated by services and information.

Superimposed on this smooth, almost evolutionary trend are some more complicated phases around the time of the Civil War and the First World War. Curiously, the Second World War has no discernible impact. The 10-year spacing of the series clearly filters out much detailed variation, but the graph retains interesting fine structure.

Very much complementary to this graph would be more standard line plots of the three sectors as time series. However, we can imagine adding a few other economies to the triangular plot without making it unreadable. The corresponding line plot with, say, a dozen time series would predictably be more difficult to read.

A major limitation on the usefulness of triangular plots as presented so far is that quite frequently data are crowded in a small part of the possible space. This is common with electoral data, especially if the third party or candidate attracts only a small percent of the total vote. Lumping all parties or candidates beyond first and second together for the sake of simplicity rarely solves this. Examples familiar to most readers are elections in the United States, long dominated by Democrats and Republicans, although often complicated in crucial ways by other parties or candidates. A variety of transformations have been suggested for triangular plots for this or other reasons. The simplest is a transformation suggested by Upton (2001). Extending `triplot` to accommodate this transformation is under way.

8 Conclusions

Many Stata users with categorical and compositional data tend to reach towards its tabulation routines in first examining datasets. This column has sampled only a few graphical possibilities in this field, some familiar staples and some possibly unfamiliar novelties, all of which are often more useful than is sometimes appreciated. Yet other graphical types have been proposed (Friendly 2000), and further Stata implementations in this territory may confidently be expected.

In the next column, the major theme will be comparison. How do we compare two or more subsets or variables, given various expectations, of equality, of additive and multiplicative shifts, and so forth? Once more, a variety of graphical types, tricks, and tips will be on display.

9 Acknowledgments

Elizabeth Allred, Ronán Conroy, Bob Fitzgerald, Roger Harbord, Friedrich Huebler, David Schwappach, Martyn Sherriff, Vince Wiggins, and Fred Wolfe made helpful comments during development of some programs discussed here. David Clayton, Anthony Edwards, and Graham Upton provided interesting discussions of triangular plots, which are likely to bear more fruit in future work on `triplot`.

10 References

Agresti, A. 2002. *Categorical Data Analysis*. Hoboken, NJ: John Wiley & Sons.

Aitchison, J. 1986. *The Statistical Analysis of Compositional Data*. London: Chapman & Hall.

Aitkin, M., D. Anderson, B. Francis, and J. Hinde. 1989. *Statistical Modelling in GLIM*. Oxford: Oxford University Press.

Atkinson, A. C. 1985. *Plots, Transformations, and Regression*. Oxford: Oxford University Press.

Beniger, J. R. 1986. *The Control Revolution: Technological and Economic Origins of the Information Society*. Cambridge, MA: Harvard University Press.

Bentley, J. L. 1988. *More Programming Pearls: Confessions of a Coder*. Reading, MA: Addison–Wesley.

Bentley, M. J., D. G. Hodgson, J. A. Smith, and N. J. Cox. Forthcoming. Preliminary relative sea level curves for the South Shetlands and Marguerite bay regions, Antarctic Peninsula. *Quaternary Science Reviews*.

Blasius, J. and M. Greenacre, ed. 1998. *Visualization of Categorical Data*. San Diego: Academic Press.

Bross, I. D. J. 1958. How to use ridit analysis. *Biometrics* 14: 38–58.

Cleveland, W. S. 1993. *Visualizing Data*. Summit, NJ: Hobart Press.

—. 1994. *The Elements of Graphing Data*. Summit, NJ: Hobart Press.

Cox, D. R. and E. J. Snell. 1989. *Analysis of Binary Data*. London: Chapman & Hall.

Cox, N. J. 1999. gr41: Distribution function plots. *Stata Technical Bulletin* 51: 12–16. In *Stata Technical Bulletin Reprints*, vol. 9, 108–112. College Station, TX: Stata Press.

—. 2001. *Plotting graded data: a Tukey-ish approach*. Presentation to UK Stata Users Group meeting, Royal Statistical Society, London, 14–15 May. Downloadable from *http://www.stata.com/support/meeting/7uk/cox1.pdf*.

—. 2002. Speaking Stata: How to face lists with fortitude. *Stata Journal* 2(2): 202–222.

—. 2003a. Software update: gr41_1: Distribution function plots. *Stata Journal* 3(2): 211.

—. 2003b. Software update: gr41_2: Distribution function plots. *Stata Journal* 3(4): 449.

—. 2004. Speaking Stata: Graphing distributions. *Stata Journal* 4(1): 66–88.

Danuso, F. 1991. gr5: Triangle graphic for soil texture. *Stata Technical Bulletin* 2: 9–10. In *Stata Technical Bulletin Reprints*, vol. 1, 40–41. College Station, TX: Stata Press.

Diaconis, P. 1988. *Group Representations in Probability and Statistics*. Hayward, CA: Institute of Mathematical Statistics.

Duncan, O. D., H. Schuman, and B. Duncan. 1973. *Social Change in a Metropolitan Community*. New York: Russell Sage Foundation.

Emerson, J. D. 1991. Introduction to transformation. In *Fundamentals of Exploratory Analysis of Variance*, ed. D. C. Hoaglin, F. Mosteller, and J. W. Tukey, 365–400. New York: John Wiley & Sons.

Fienberg, S. E. 1980. *The Analysis of Cross-classified Categorical Data*. Cambridge, MA: MIT Press.

Fisher, R. A. 1940. The precision of discriminant functions. *Annals of Eugenics* 10: 422–429.

Fleiss, J. L., B. Levin, and M. C. Paik. 2003. *Statistical Methods for Rates and Proportions*. Hoboken, NJ: John Wiley & Sons.

Flora, J. D. 1988. Ridit analysis. In *Encyclopedia of Statistical Sciences*, ed. S. Kotz and N. L. Johnson, vol. 8, 136–139. New York: John Wiley & Sons.

Friendly, M. 2000. *Visualizing Categorical Data*. Cary, NC: SAS Institute.

Gower, J. C. and D. J. Hand. 1996. *Biplots*. London: Chapman & Hall.

Gray, J. 1993. Möbius's geometrical mechanics. In *Möbius and his Band: Mathematics and Astronomy in Nineteenth-century Germany*, ed. J. Fauvel, R. Flood, and R. Wilson, 79–103. Oxford: Oxford University Press.

Haberman, S. J. 1996. *Advanced Statistics Volume I: Description of Populations*. New York: Springer.

Howarth, R. J. 1996. Sources for a history of the ternary diagram. *British Journal for the History of Science* 29: 337–356.

Knoke, D. and P. J. Burke. 1980. *Log-linear Models*. Beverly Hills, CA: Sage.

Lloyd, C. J. 1999. *Statistical Analysis of Categorical Data*. New York: John Wiley & Sons.

Marden, J. I. 1995. *Analyzing and Modeling Rank Data*. London: Chapman & Hall.

McCullagh, P. and J. A. Nelder. 1989. *Generalized Linear Models*. 2d ed. London: Chapman & Hall.

Mitchell, M. 2004. *A Visual Guide to Stata Graphics*. College Station, TX: Stata Press.

Möbius, A. F. 1827. *Der barycentrische Calcul: ein neues Hülfsmittel zur analytischen Behandlung der Geometrie dargestellt und insbesondere auf die Bildung neuer Classen von Aufgaben und die Entwicklung mehrerer Eigenschaften der Kegelschnitte*. Leipzig: Johann Ambrosius Barth.

Parzen, E. 1993. Change *PP* plot and continuous sample quantile function. *Communications in Statistics – Theory and Methods* 22: 3287–3304.

Royston, P. 1996. gr21: Flexible axis scaling. *Stata Technical Bulletin* 34: 9–10. In *Stata Technical Bulletin Reprints*, vol. 6, 34–36. College Station, TX: Stata Press.

Simonoff, J. S. 2003. *Analyzing Categorical Data*. New York: Springer.

Stevens, S. S. 1946. On the theory of scales of measurement. *Science* 103: 677–680.

Stouffer, S. A., A. A. Lumsdaine, M. H. Lumsdaine, R. M. Williams, M. B. Smith, I. L. Janis, S. A. Star, and L. S. Cottrell. 1949. *The American Soldier: Combat and its Aftermath*. Princeton, NJ: Princeton University Press.

Tocher, J. F. 1908. Pigmentation survey of school children in Scotland. *Biometrika* 6: 129–235.

Townsend, J. G. 1995. *Women's Voices from the Rainforest*. London: Routledge.

Tukey, J. W. 1960. The practical relationship between the common transformations of percentages or fractions and of amounts. Reprinted in *The Collected Works of John W. Tukey. Volume VI: More Mathematical*, 1990, ed. C. L. Mallows, 211–219. Pacific Grove, CA: Wadsworth and Brooks/Cole.

—. 1961. Data analysis and behavioral science or learning to bear the quantitative man's burden by shunning badmandments. Reprinted in *The Collected Works of John W. Tukey. Volume III: Philosophy and Principles of Data Analysis: 1949–1964*, 1986, ed. L. V. Jones, 187–389. Monterey, CA: Wadsworth and Brooks/Cole.

—. 1977. *Exploratory Data Analysis*. Reading, MA: Addison–Wesley.

Upton, G. J. G. 2001. A toroidal scatter diagram for ternary variables. *The American Statistician* 55: 247–250.

Wild, C. J. and G. A. F. Seber. 2000. *Chance Encounters: A First Course in Data Analysis and Inference*. New York: John Wiley & Sons.

About the Author

Nicholas Cox is a statistically minded geographer at the University of Durham. He contributes talks, postings, FAQs, and programs to the Stata user community. He has also co-authored fourteen commands in official Stata. He was an author of several inserts in the *Stata Technical Bulletin* and is Executive Editor of the *Stata Journal*.