The Stata Journal (2016)
16, Number 2, pp. 491–510

Speaking Stata: Multiple bar charts in table form

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

Abstract. Tables that are one way, two way, or three way in structure may often be helpfully represented as multiple bar charts. The one, two, or three variables that define the structure of the table thus determine rows, columns, and panels in which bars are arranged. The merits of this design include easy focus on individual values or groups of values; leaving space for numeric information to be shown as in a table; and convenient axis or panel labeling through text rather than through a key or legend. A Stata command for these purposes, `tabplot`, is discussed systematically.

Keywords: gr0066, tabplot, bar charts, histograms, mosaic plots, spineplots, graphics, tables, categorical data

1 Introduction

Articles in this journal often introduce new Stata commands. This column introduces an *old* Stata command, which goes back to 1999. `tabplot` for plotting multiple bar charts has been discussed briefly in previous columns (Cox 2004, 2008b, 2012). Here I discuss the main ideas more systematically. Without contradiction or paradox, one may say that while the leading idea behind `tabplot` has been well known in many sciences for decades, it is nevertheless neglected in several others, and possibly even unknown in some quarters.

"Multiple bar charts" is one simple umbrella term that covers what `tabplot` can do. However, that does not mean stacked or divided bars, and it does not mean bars side by side on the same axis. A lengthier but less ambiguous explanation might be one-, two- or three-way bar charts in table form.

Why do we need another bar chart command in Stata? Bar charts are basic and may seem well supported in Stata. Only a little acquaintance with the manual documentation reveals four main official commands, `graph bar`, `graph hbar`, `twoway bar`, and `twoway rbar`. That might seem already three more than one might need. Indeed, `tabplot` is just a wrapper for `twoway rbar`. However, it can produce various plots more easily and more quickly than you could do yourself, unless you were willing to do a little programming and a lot of fiddling around.

In a nutshell, the main conceit of `tabplot` is tablelike plots. The name is intended to evoke commands like `tabulate` with their structured output of tables in rows and columns.

Section 2 dives straight in with two substantial examples of `tabplot`. Section 3 discusses other features, and section 4 links to previous literature. Section 5 gives a different kind of example. Section 6 is a more formal statement of `tabplot` syntax.

2 Illustrations

2.1 Two-way table example

Greenacre (2007, 42) published the dataset we will use for the first example. The data come from the 1997 Encuesta Nacional de la Salud (Spanish National Health Survey). They are interesting in themselves, but for present purposes, they are useful as an example simple enough not to require specialist knowledge but large enough to be a little challenging. As with many tables, the main handle for understanding is to look at the probability distribution of the response, which is here an ordered (ordinal, graded) variable, `health`, given a predictor, here another ordered variable, `agegroup`. As often, a cross-sectional dataset is partly of interest for what it might convey indirectly about longitudinal patterns.

`tabplot` offers options to calculate percent or proportional (fractional) breakdowns on the fly. Aesthetic preferences or social conventions often encourage presentation in terms of percents rather than proportions. ("Percentage" seems to me too long a word, whatever dictionaries may say.)

```
clear
input byte(agegroup health) long freq
1 1 243
1 2 789
1 3 167
1 4 18
1 5 6
2 1 220
2 2 809
2 3 164
2 4 35
2 5 6
3 1 147
3 2 658
3 3 181
3 4 41
3 5 8
4 1 90
4 2 469
4 3 236
4 4 50
4 5 16
5 1 53
5 2 414
5 3 306
```

```
5 4 106
5 5 30
6 1 44
6 2 267
6 3 284
6 4 98
6 5 20
7 1 20
7 2 136
7 3 157
7 4 66
7 5 17
end
label values agegroup agegroup
label def agegroup 1 "16-24", modify
label def agegroup 2 "25-34", modify
label def agegroup 3 "35-44", modify
label def agegroup 4 "45-54", modify
label def agegroup 5 "55-64", modify
label def agegroup 6 "65-74", modify
label def agegroup 7 "75+", modify
label values health health
label def health 1 "very good", modify
label def health 2 "good", modify
label def health 3 "regular", modify
label def health 4 "bad", modify
label def health 5 "very bad", modify
tabplot health agegroup [w=freq], percent(agegroup) showval   ///
     subtitle(% of age group) xtitle("") bfcolor(none)
```

Figure 1 shows the result.

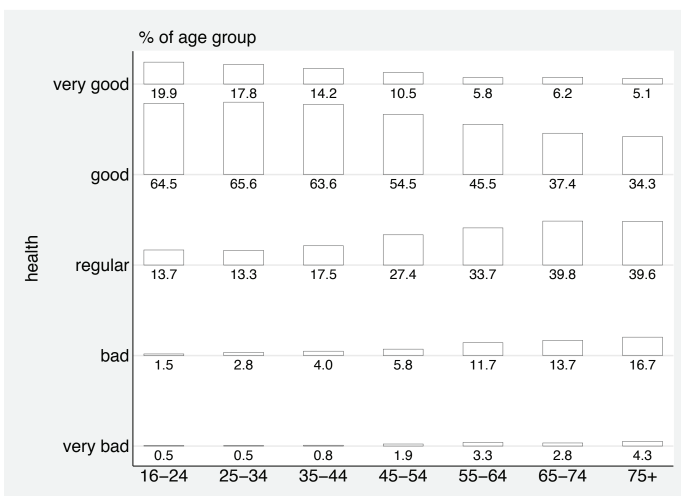

Figure 1. Two-way bar chart showing perception of health, with percent breakdown by age group according to the 1997 Encuesta Nacional de la Salud (Spanish National Health Survey). Source of data: Greenacre (2007, 42).

`tabplot` with two variables matches a standard `twoway` convention in that the first-named variable is plotted on the y axis. Usually, but not invariably, this variable is the response or outcome variable of interest. The first-named variable thus defines rows of the two-way bar chart, and the second-named variable, plotted on the x axis, defines columns of the chart.

In a simple application like this, `tabplot` counts frequencies of cross-combinations before producing a graph. As seen, it accepts frequency weights for the common case in which data are aggregated by cross-combinations of categories and frequencies are held in another variable. Although plots of raw frequencies are often of interest, it is common to show proportional or percent breakdowns, usually conditional on one or more predictor variables.

Figure 1 exemplifies a common feature with categorical data. Some categories may be relatively rare. There are some very small percents in the plot. A merit of the multiple-bar-charts design is that small values are discernible as such.

The `showval` option (think "show values") is a crucial detail of `tabplot`. Although not a default, it is likely to be used in most applications of the command. If we do insist on showing values too, then we are deliberately making use of table form as well as graph form. If percents are shown, one decimal place is the default (12.3, say). If proportions are shown, three decimal places are the default (0.123, say).

The facility to show individual values is most valuable when there is a desire to "look up" those values. This can come either before or after the graph is drawn. What is the value for such-and-such compared with others? Look, that value is a worrying 4.2%! But that 66.6% is good news!

Some researchers argue graphs are graphs and tables are tables, and ne'er the twain shall meet. An intriguing suggestion, which I have borrowed elsewhere (Cox 2003), is that the conventional distinction between graphs and tables was a side effect of the development of printing. Before printing, there were manuscripts—those scripted manually or written by hand—to which writers could freely add instructive and entertaining illustrations, whether factual, fabulous, or fantastic. Printed documents encouraged, or even enforced, a division of labor between typesetters and those who prepared illustrations. But now that division is largely obsolete. Writers and publishers have more flexibility in mixing text and illustrations of whatever kind, including not only figures and tables but also hybrid graphs and tables (see also Cox [2008a]).

Given this dataset, how else would you represent the patterns graphically? Setting aside any temptation to draw multiple pie charts, you might opt for the alternative of a stacked bar chart. Figure 2 shows one result.

```
graph bar (count) [fw=freq], over(health, descending) over(agegroup)     ///
     percent subtitle(% of age group) stack asyvars bar(5, bcolor(gs1))  ///
     bar(4, bcolor(gs4)) bar(3, bcolor(gs7)) bar(2, bcolor(gs10))        ///
     bar(1, bcolor(gs14)) legend(pos(3) column(1)) yscale(r(-5 100))     ///
     ylabel(, angle(h))
```

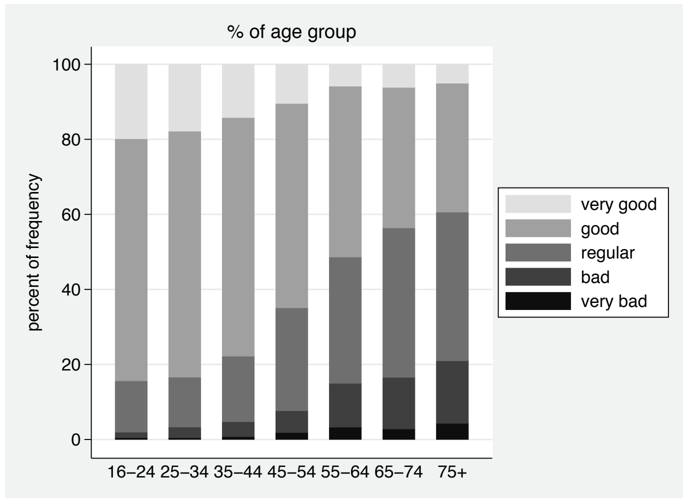

Figure 2. Stacked bar chart showing perception of health, with percent breakdown by age group according to the 1997 Encuesta Nacional de la Salud (Spanish National Health Survey). Source of data: Greenacre (2007, 42).

This example shows the constraints of printing in black and white. You might well prefer to experiment without that constraint. Here is one command with less inhibited use of color, which you can try for yourself if interested:

```
graph bar (count) [fw=freq], over(health, descending) over(agegroup)        ///
     percent subtitle(% of age group) stack asyvars                         ///
     bar(5, bfcolor(red*0.8)) bar(4, bfcolor(red*0.3) blcolor(red*0.8))     ///
     bar(3, bfcolor(blue*0.2) blcolor(blue*1.2))                            ///
     bar(2, bfcolor(blue*0.7) blcolor(blue*1.2)) bar(1, bcolor(blue*1.2))   ///
     legend(position(3) column(1)) yscale(range(-5 100)) ylabel(, angle(h))
```

Another graph that is popular in some fields is a mosaic plot, or more specifically a spineplot, here made using the `spineplot` command (Cox 2008b, 2016). Figure 3 is one such plot.

```
spineplot health age [w=freq], xlabel(, labsize(*0.8) axis(2)) percent      ///
     xlabel(0(20)100, tposition(outside)) ylabel(0(20)100, axis(2))         ///
     bar1(bcolor(gs1)) bar2(bcolor(gs4)) bar3(bcolor(gs7))                  ///
     bar4(bcolor(gs10)) bar5(bcolor(gs14)) barall(blwidth(none))
```

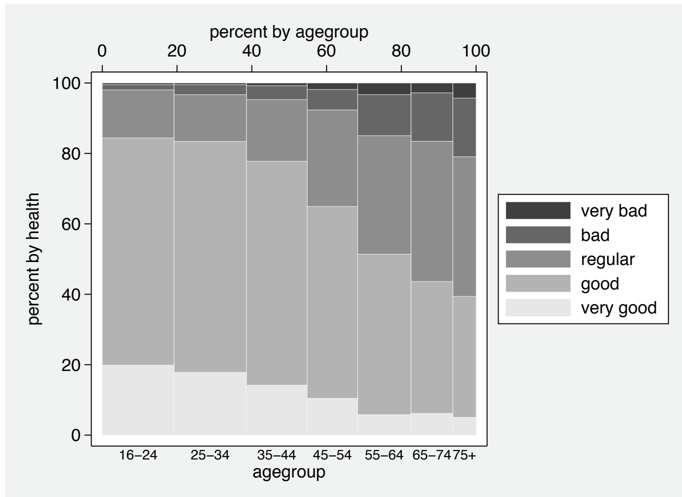

Figure 3. Spineplot (mosaic plot) showing perception of health, with percent breakdown by age group according to the 1997 Encuesta Nacional de la Salud (Spanish National Health Survey). Source of data: Greenacre (2007, 42).

As often emphasized, spineplots have various distinctive advantages. They faithfully show raw frequencies as well as row and column proportions. The area of each tile is proportional to absolute frequency, here the number of people in each cross-combination. The vertical and horizontal sides of each tile represent the proportions defined by the row and column variables. Collectively, the pattern of tiles shows the relationship between the variables. Independence of variables would imply a simple pattern in which row tiles are aligned, just as column tiles are. Departures from independence show departures from that benchmark.

Conversely, the spineplot design necessarily implies that tiles may be very small (or even not exist). In figure 3, three small tiles are occluded completely. However, that could be seen as a fault of the implementation of `spineplot`: in some other software, it is common to insert small spaces between tiles for readability.

In general, however, spineplots are based on the principle that tile areas represent absolute or relative frequencies. `tabplot`, like most bar chart programs, shows bars equal in width or height in one dimension and so is based on the simpler principle that those frequencies are encoded by bar heights or lengths in the other dimension.

So, which graphs are better (say, clearer, more effective, more attractive)? People can easily disagree here, depending partly on personal experience and taste and partly on ideas about what the readers of any graph will appreciate. There is often resistance to adopting unfamiliar designs. Sometimes that disinclination seems to imply that you must never show a graph that readers may not have seen before, which raises the question of how anyone ever learned about a graph for the first time.

Stacking is a well-understood design, but very small amounts are hard to spot on a stacked graph. Similarly, any legend obliges mental "back and forth" from readers (or else they give up on looking at the detail). Both `graph bar` and `spineplot` let you add numeric values on top of the bars or tiles. However, for this example, and for many others with some small amounts to show, that would be at least a little messy. The tradeoff is difficult to get right: different colors or shadings are essential to distinguish the different bars in stacked graphs, but that undermines showing numeric values too.

2.2 Three-way table example

Aitkin et al. (1989, 242) reported data from a survey of student opinion on the Vietnam War taken at the University of North Carolina at Chapel Hill in May 1967. Students were classified by sex, year of study, and the policy they supported, given the following choices:

A. The United States should defeat the power of North Vietnam by widespread bombing of its industries, ports, and harbors and by land invasion.

B. The United States should follow the present policy in Vietnam.

C. The United States should de-escalate its military activity, stop bombing North Vietnam, and intensify its efforts to begin negotiation.

D. The United States should withdraw its military forces from Vietnam immediately.

The labels A through D are charmless, but even at this distance, suggesting better ones might be thought contentious politically, so I will let them stand. What is important is that the sequence is ordered, from "hawk" to "dove" in contemporary terms.

```
clear
input str6 sex str8 year str1 policy int freq
"male" "1" "A" 175
"male" "1" "B" 116
"male" "1" "C" 131
"male" "1" "D" 17
"male" "2" "A" 160
"male" "2" "B" 126
"male" "2" "C" 135
"male" "2" "D" 21
"male" "3" "A" 132
"male" "3" "B" 120
"male" "3" "C" 154
"male" "3" "D" 29
"male" "4" "A" 145
"male" "4" "B" 95
"male" "4" "C" 185
"male" "4" "D" 44
"male" "Graduate" "A" 118
"male" "Graduate" "B" 176
"male" "Graduate" "C" 345
```

```
"male" "Graduate" "D" 141
"female" "1" "A" 13
"female" "1" "B" 19
"female" "1" "C" 40
"female" "1" "D" 5
"female" "2" "A" 5
"female" "2" "B" 9
"female" "2" "C" 33
"female" "2" "D" 3
"female" "3" "A" 22
"female" "3" "B" 29
"female" "3" "C" 110
"female" "3" "D" 6
"female" "4" "A" 12
"female" "4" "B" 21
"female" "4" "C" 58
"female" "4" "D" 10
"female" "Graduate" "A" 19
"female" "Graduate" "B" 27
"female" "Graduate" "C" 128
"female" "Graduate" "D" 13
end
tabplot policy year [w=freq], by(sex, subtitle(% by sex and year, place(w)) ///
     note("")) percent(sex year) showval
```

Figure 4 shows a three-way bar chart.

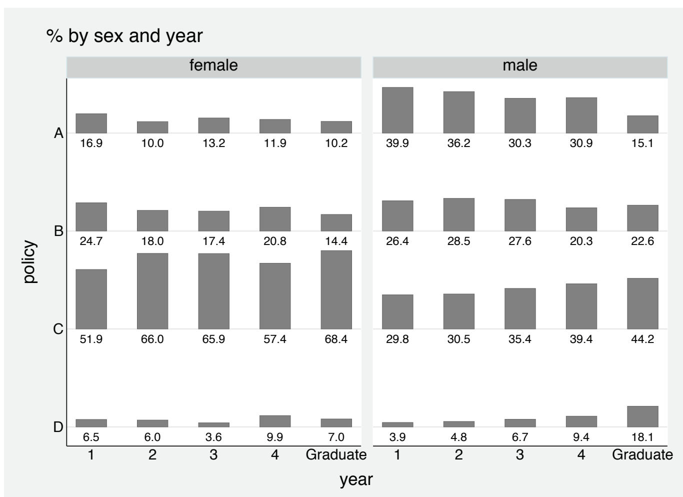

Figure 4. Three-way bar chart showing student preferences for policy options in the Vietnam War, University of North Carolina at Chapel Hill in May 1967, with percent breakdown by sex and year of study. Source of data: Aitkin et al. (1989, 242).

As with the previous example, the data are cross-sectional, not longitudinal, however tempting an interpretation may be in terms of students changing their opinions over time as they get older.

The way to plot three-way tables is by using a `by()` option to repeat two-way tables. As with two-way tables, it is usually best to specify the response or outcome variable first, as defining rows of the plot and as to be shown on the y axis. The order of the other two variables is at choice. There can be tradeoffs or compromises, because no layout is best for all purposes. Big differences can safely be put at a distance (so males and females here differ markedly in their mix of views), while finer comparisons are easier to make if bars are close. On top of all that, any ordinal scales should naturally be respected as such. A wider issue, not addressed directly here, is the scope for correspondence analysis or some other multivariate method to choose a good ordering for other sets of categories, an approach known in archeology and other sciences as seriation (Hahsler, Hornik, and Buchta 2008). This was precisely the machinery behind table 1 in Roberts et al. (2013).

As with the previous example, are there graph forms that make the data easier to think about?

3 More features

One-way bar charts are also possible with `tabplot`. The main reason for using them is if `graph bar`, `graph hbar`, and `histogram` do not satisfy, particularly if you prefer the display provided by the `showval()` option.

We have shown the default of vertical bars. A simple alternative is to use the `horizontal` option to get horizontal bars. In that case, it is usually easier to compare down columns. Some experimentation with both forms will often be helpful.

`tabplot` *varname*`, by()` is another way to plot two-way tables.

Four-way or higher charts would often not be readable or interpretable, but there are three evident ways to attempt them. First, try to `reshape` or otherwise restructure the data concerned to fewer variables. Second, combine variables, usually predictor variables, into a composite variable to be shown on one axis. See Cox (2007) for discussion of how to do that. Third, use `tabplot` repeatedly and then use `graph combine`.

`tabplot` with the `xasis` option may be useful for stacking histograms vertically. Less commonly, with the `yasis` and `horizontal` options, it may be useful for stacking them horizontally. The help file gives examples. In general, specify a variable containing equally spaced midpoints, and assign to it an appropriate variable label. `tabplot` will do the rest. Omit the `percent()` option for display of frequencies.

A method for subverting `tabplot` to plot any variable that takes on a single value for each cross-combination of categories is illustrated in the examples in the help file. The key is to select precisely one observation for each cross-combination and to specify that variable as (most generally) an `iweight`.

Furthermore, using an `iweight` is the only possible method whenever a variable has at least some negative values. In that case, you might do the following:

1. Consider changing the maximum height with `height()` to avoid overlap of bars variously representing positive and negative values. By default, `tabplot` chooses the scale to accommodate the longest bar to be shown, but it contains no special intelligence otherwise to avoid overlap of bars in the same column or row.
2. If also using `showval` or `showval()`, consider changing `offset()` and using a transparent `bfcolor()`.

Yet further variants can be specified through other options.

No example here shows a large table, partly because you need a big monitor or a bigger paper size than we have in the *Stata Journal* to do one justice. I have had reasonable results with `tabplot` and tables on the order of a thousand cells. The emphasis is then typically on overall patterns, although striking details can sometimes be seen. `showval()` is usually out of the question. For larger tables, other tricks can be helpful, such as suppressing small amounts or truncating large amounts using the `minimum()` or `maximum()` option; coloring different kinds of bars differently; and showing bar counts or amounts on some transformed scale, typically a square-root scale. The last is achieved by transforming before the program is called and passing the transformed amounts as importance weights.

4 Remarks on the literature

Bar charts presented as one row or one column of bars go back at least as far as Playfair (1786). See, for example, Playfair (2005, 25) or Wainer (2005, 45; 2009, 174).

Bar charts presented in table form with two or more rows and two or more columns are less common. They have been used in one form of pollen diagram. Sears (1933, 1935) gave some early examples. See also Emeny (1934) for a well-illustrated monograph on raw materials. The help file for `tabplot` gives many other references. Among more recent work, the examples and exhortations of Bertin (1981, 1983) are especially notable. See also Morrison (1985), de Falguerolles, Friedrich, and Sawitzki (1997), and Chauchat and Risson (1998) for articles influenced by Bertin.

As the example of pollen diagrams shows, the same form of graph can be used for showing on any one axis either the categories of what is regarded as one variable or two or more variables considered similar or comparable. Such bar charts, or similar displays, are also known as

- aligned bar charts and multipane bar charts (see Mackinlay [1986] and McDaniel and McDaniel [2012a; 2012b]);
- survey plots (see Lohninger [1994; 1996], Hoffman and Grinstein [2002], Grinstein et al. [2002], and Ward, Grinstein, and Keim [2010]);

- table lens (Rao and Card [1994], Pirolli and Rao [1996], Spence [2007], Ward, Grinstein, and Keim [2010], and Few [2012]); and

- multiple bar charts (as here) and fluctuation diagrams (see Becker, Chambers, and Wilks [1988], Unwin, Theus, and Hofmann [2006], Hofmann [2008], Theus and Urbanek [2009], and Unwin [2015]).

Such bar charts may require no more than `reshape`. Shortly, I will give an example with archaeological data.

Displays such as bar and pie charts with added numeric labels have been called "grables" (Hink, Wogalter, and Eustace 1996; Hink, Eustace, and Wogalter 1998; Bradstreet 2012). Somehow, this term has not proved widely attractive.

Note also what are often called Hinton diagrams or Hinton plots in machine learning. Rumelhart, Hinton, and Williams (1986) is a token reference. Examples occur in mainstream machine-learning texts such as MacKay (2003), Bishop (2006), Barber (2012), and Murphy (2012).

Brinton (1939, 142, 505) uses the term "two-way bar chart" for back-to-back or bilateral bar charts, a use different from that here.

5 A different kind of example

Doran and Hodson (1975, 259) gave these archaeological data from the rock shelter of Ksar Akil, near Beirut, which relate to the Upper Paleolithic. The data are counts of three kinds of artifacts. Although the data arrive as three count variables together with level, an indicator of depth (highest-numbered level is uppermost), it is easy to reshape them to a two-way table.

```
clear
input levels freqcores freqblanks freqtools
    25 21 32 70
    24 36 52 115
    23 126 650 549
    22 159 2342 1633
    21 75 487 511
    20 176 1090 912
    19 132 713 578
    18 46 374 266
    17 550 6182 1541
    16 76 846 349
    15 17 182 51
    14 4 51 14
    13 29 228 130
    12 135 2227 729
end
reshape long freq, i(levels) j(type) string
tabplot levels type [w=freq], bfcolor(none) horizontal barwidth(1)      ///
     percent(levels) subtitle(% at each level) showval(offset(0.45))    ///
     xscale(r(0.8 .)) yasis
```

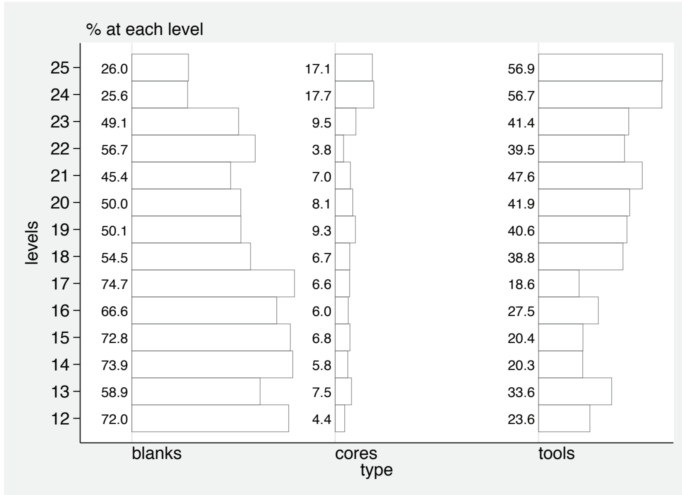

Figure 5. Two-way bar chart showing variations in relative abundance of cores, blanks, and tools from various levels at Ksar Akil. Source of data: Doran and Hodson (1975, 259).

Figure 5 shows how other choices can seem natural, or at least convenient. In archaeology, and also commonly in the earth and environmental sciences, variations with depth below (or height above) the surface are commonly plotted with depth or height on the y axis. That is how data were collected, and it is how researchers often think about their problems (see Cox and Barlow [2008] for another example).

Given a choice of depth for the y axis, and the `yasis` option to enforce a literal reading of levels, horizontal bars are called for. The percents go to the left of each bar, and for variety, as well as logic, the bars are made to touch using the `barwidth()` option. A subtle default is that axis ticks are explicit with the `yasis` or `xasis` option, the inference being that the scale is measured or counted, so that ticks are then natural. That is just the default, and it can be overruled.

6 The tabplot command

6.1 Syntax

`tabplot` *varname* [*if*] [*in*] [*weight*] [, *options*]

`tabplot` *rowvar colvar* [*if*] [*in*] [*weight*] [, *options*]

`fweight`s, `aweight`s, and `iweight`s may be specified.

See section 6.3 for the available *options*.

6.2 Description

`tabplot` plots a table of numerical values (for example, frequencies, fractions, or percents) in graphical form as a bar chart. It is mainly intended for representing contingency tables for one, two, or three categorical variables. It also has uses for producing multiple histograms and graphs for general one-, two-, or three-way tables.

`tabplot` *varname* creates a bar chart that by default displays one set of vertical bars; with the `horizontal` option, it displays one set of horizontal bars. The categories of *varname* thus define either columns from left (low values) to right (high values) or rows from top (low values) to bottom (high values). The value (for example, frequency, fraction, or percent) for each column or row is shown as a bar.

`tabplot` *rowvar colvar* follows standard tabular alignment: the categories of *rowvar* define rows from top (low values) to bottom (high values), and the categories of *colvar* define columns from left (low values) to right (high values). The value (for example, frequency, fraction, or percent) for each combination of row and column is shown as a bar with default alignment vertical.

The default bar width is 0.5. Use the `barwidth()` option to vary width, but note that all bars will have the same width.

By default, both variables are mapped on the fly in sort order to successive integers from 1 and up, but original values or value labels are used as value labels: this may be varied by use of the `yasis` or `xasis` option. The maximum bar height is by default 0.8. Use the `height()` option to vary this.

6.3 Options

`fraction` indicates that all frequencies should be shown as fractions (with sum 1) of the total frequency of all values being represented in the graph.

`fraction(`*varlist*`)` indicates that all frequencies should be shown as fractions (with sum 1) of the total frequency for each distinct category defined by the combinations of *varlist*. Usually, *varlist* will be one or more of the variables specified.

`percent` indicates that all frequencies should be shown as percents (with sum 100) of the total frequency of all values being represented in the graph.

`percent(`*varlist*`)` indicates that all frequencies should be shown as percents (with sum 100) of the total frequency for each distinct category defined by the combinations of *varlist*. Usually, *varlist* will be one or more of the variables specified.

Only one of the `fraction`[`()`] and `percent`[`()`] options may be specified.

`missing` specifies that any missing values of any of the variables specified should also be included within their own categories.

`yasis` and `xasis` specify, respectively, that the y (row) variable and the x (column) variable are to be treated literally (that is, numerically). Most commonly, each option will be specified if the variable in question is a measured scale or a graded variable with gaps. If values 1 to 5 are labeled A to E, but no value of 4 (D) is present in the data, `yasis` or `xasis` prevents a mapping to 1 (A) ... 4 (E).

`horizontal` specifies horizontal bars. The default is vertical bars.

`height(`#`)` controls the amount of graph space taken up by bars. The default is `height(0.8)`. Note that the height may need to be much smaller or much larger with `yasis` or `xasis`, given that the latter take values literally.

`showval` specifies that numeric values be shown beneath (or if `horizontal` is specified, to the left of) bars.

`showval` may also be specified with a variable name and options. If options alone are specified, no comma is necessary. In particular,

`showval(`*varname*`)` specifies that the values to be shown are those of *varname*. For example, the values of some kind of residuals might be shown alongside frequency bars.

`showval(offset(`#`))` specifies an offset between the base (or left-hand edge) of the bar and the position of the numeric value. The default is 0.1 with two variables or 0.02 with one variable. Tweak this if the spacing is too large or too small.

`showval(format(`*format*`))` specifies a format with which to show values. Specifying a format will often be advisable with nonintegers; for example, using `showval(format(%2.1f))` specifies rounding to 1 decimal place. Note that with a specified variable, the format defaults to the format of that variable; with percent options, the format defaults to %2.1f (1 decimal place); with fraction options, the format defaults to %4.3f (3 decimal places).

`showval(`*varname*`, format(%2.1f))` is an example of *varname* specified with options. As usual, a comma is needed in such cases.

Otherwise, the options of `showval()` can be options of `scatter` (see [G-2] **graph twoway scatter**), most usually marker label options.

`minimum(#)` suppresses plotting of bars with values less than the minimum specified, in effect setting them to zero.

`maximum(#)` truncates bars with values more than the maximum specified to show that maximum.

`separate(`*sepspec*`)` specifies that bars associated with different *sepspec* will be shown differently, most obviously using different colors. *sepspec* is passed as an argument to the `by()` option of `separate`, except that references to `@` are first translated to be references to the quantity being plotted.

A call to `separate()` may be supplemented with calls to options `bar1()` ... `bar20()` or to `barall()`. The arguments should be options of `twoway rbar`.

`bar1(`*rbar_options*`)` ... `bar20(`*rbar_options*`)` are provided to allow overriding the defaults on up to 20 categories, the first, second, etc., shown. The limit of 20 is arbitrary and more than any user should really want.

`barall(`*rbar_options*`)` overrides the defaults for all bars. Any `bar#()` option always overrides `barall()`. Thus if you wanted thicker `blwidth()` on all bars, you could specify `barall(blwidth(thick))`. If you wanted to highlight the first category only, you could specify `bar1(blwidth(thick))`.

graph_options refers to options of `twoway rbar`; see [G-2] **graph twoway rbar**. Among others:

`barwidth(#)` specifies the widths of the bars. The default is `barwidth(0.5)`. This may need changing, especially with the `xasis` or `yasis` option or if you wish bars to touch, exactly or nearly.

`bfcolor(`*colorstyle*`)` adjusts the bar fill color. In particular, Stata's defaults often imply that bars are filled with strong colors, but unfilled bars created by using `bfcolor(none)` may be more subtle and just as clear.

`by(`*varlist*`)` specifies another variable used to subdivide the display into panels.

`recast(`*newplottype*`)` recasts the graph as another twoway plottype. In practice, `recast(rspike)` is the main alternative.

`subtitle(`*tinfo*`)`, shown by default outside the graph and at top left, specifies what kind of quantity is being shown: `"frequency"`, `"percent"`, and so forth. The examples in the help file show how the subtitle is changed, which may mean being blanked out.

`plot(`*plots*`)` provides a way to add other plots to the generated graph. Allowed in Stata 8.

`addplot(`*plots*`)` provides a way to add other plots to the generated graph. Allowed in Stata 9 and later.

With large datasets especially, one should ensure that the plot or the extra plots do not contain information repeated for every observation within each combination of *rowvar* and *colvar*. The examples in the help file show one technique for avoiding this.

7 Conclusions

`tabplot` implements a simple twist on bar charts that can impart helpful spin. By separating bars, yet preserving table layout, we allow focus on individual values as well as overview of bars as a collective pattern. This can be especially useful for two- or three-dimensional tables. There is, at least if tables are not too large, space for numeric information to be shown, just as in a table. A further small virtue is convenient axis or panel labeling through text rather than through a key or legend.

8 Acknowledgments

This column is a personal salute to two very different people. Jacques Bertin (1918–2010) was a leader in cartography and visualization, a maverick (and even mischievous) maestro of the graphic. Bar charts in table form were among his recommended designs. Sir David MacKay (1967–2016) was enormously original and influential in a range from machine learning, Bayesian style, to sustainable energy policy. The illustrations in his books (MacKay 2003, 2009) are miniatures of economy and elegance.

Bob Fitzgerald, Friedrich Huebler, and Martyn Sherriff found typos in earlier versions of the `tabplot` help file. Friedrich also pointed to various efficiency issues. Marcello Pagano provided encouragement and found a bug. Vince Wiggins suggested how best to align x-axis labels when bars are horizontal. Jay Goodliffe suggested flagging use of the `subtitle()` option in the help.

9 References

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

Barber, D. 2012. *Bayesian Reasoning and Machine Learning*. Cambridge: Cambridge University Press.

Becker, R. A., J. M. Chambers, and A. R. Wilks. 1988. *The New S Language: A Programming Environment for Data Analysis and Graphics*. Pacific Grove, CA: Wadsworth and Brooks/Cole.

Bertin, J. 1981. *Graphics and Graphic Information Processing*. Berin: De Gruyter.

———. 1983. *Semiology of Graphics: Diagrams, Networks, Maps*. Madison: University of Wisconsin Press.

Bishop, C. M. 2006. *Pattern Recognition and Machine Learning*. New York: Springer.

Bradstreet, T. E. 2012. Grables: Visual displays that combine the best attributes of graphs and tables. In *A Picture is Worth a Thousand Tables: Graphics in Life Sciences*, ed. A. Krause and M. O'Connell, 41–69. New York: Springer.

Brinton, W. C. 1939. *Graphic Presentation*. New York: Brinton Associates.

Chauchat, J.-H., and A. Risson. 1998. Bertin's graphics and multidimensional data analysis. In *Visualization of Categorical Data*, ed. J. Blasius and M. Greenacre, 37–45. San Diego, CA: Academic Press.

Cox, N. J. 2003. Speaking Stata: Problems with tables, Part I. *Stata Journal* 3: 309–324.

———. 2004. Speaking Stata: Graphing categorical and compositional data. *Stata Journal* 4: 190–215.

———. 2007. Stata tip 52: Generating composite categorical variables. *Stata Journal* 7: 582–583.

———. 2008a. Speaking Stata: Between tables and graphs. *Stata Journal* 8: 269–289.

———. 2008b. Speaking Stata: Spineplots and their kin. *Stata Journal* 8: 105–121.

———. 2012. Speaking Stata: Axis practice, or what goes where on a graph. *Stata Journal* 12: 549–561.

———. 2016. Software Updates: Spineplots and their kin. *Stata Journal* 16: 521–522.

Cox, N. J., and N. L. M. Barlow. 2008. Stata tip 62: Plotting on reversed scales. *Stata Journal* 8: 295–298.

de Falguerolles, A., F. Friedrich, and G. Sawitzki. 1997. A tribute to J. Bertin's graphical data analysis. In *Softstat '97: Advances in Statistical Software 6: The 9th Conference on the Scientific Use of Statistical Software, March 3–6, 1997*, ed. W. Bandilla and F. Faulbaum, 11–20. Stuttgart: Lucius & Lucius.

Doran, J. E., and F. R. Hodson. 1975. *Mathematics and Computers in Archaeology*. Edinburgh: Edinburgh University Press.

Emeny, B. 1934. *The Strategy of Raw Materials: A Study of America in Peace and War*. New York: Macmillan.

Few, S. 2012. *Show Me the Numbers: Designing Tables and Graphs to Enlighten*. 2nd ed. Burlingame, CA: Analytics Press.

Greenacre, M. 2007. *Correspondence Analysis in Practice*. 2nd ed. Boca Raton, FL: Chapman & Hall/CRC.

Grinstein, G. G., P. E. Hoffman, R. M. Pickett, and S. J. Laskowski. 2002. Benchmark development for the evaluation of visualization for data mining. In *Information Visualization in Data Mining and Knowledge Discovery*, ed. U. Fayyad, G. G. Grinstein, and A. Wierse, 129–176. San Diego, CA: Academic Press.

Hahsler, M., K. Hornik, and C. Buchta. 2008. Getting things in order: An introduction to the R package seriation. *Journal of Statistical Software* 25(3): 1–34. https://www.jstatsoft.org/article/view/v025i03.

Hink, J. K., J. K. Eustace, and M. S. Wogalter. 1998. Do grables enable the extraction of quantitative information better than pure graphs or tables? *International Journal of Industrial Ergonomics* 22: 439–447.

Hink, J. K., M. S. Wogalter, and J. K. Eustace. 1996. Display of quantitative information: Are grables better than plain graphs or tables? *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 40: 1155–1159.

Hoffman, P. E., and G. G. Grinstein. 2002. A survey of visualizations for high-dimensional data mining. In *Information Visualization in Data Mining and Knowledge Discovery*, ed. U. Fayyad, G. G. Grinstein, and A. Wierse, 47–82. San Diego, CA: Academic Press.

Hofmann, H. 2008. Mosaic plots and their variants. In *Handbook of Data Visualization*, ed. C. Chen, W. Härdle, and A. Unwin, 617–642. Berlin: Springer.

Lohninger, H. 1994. INSPECT: A program system to visualize and interpret chemical data. *Chemometrics and Intelligent Laboratory Systems* 22: 147–153.

———. 1996. INSPECT*: A Program System for Scientific and Engineering Data*. Berlin: Springer.

MacKay, D. J. C. 2003. *Information Theory, Inference, and Learning Algorithms*. Cambridge: Cambridge University Press.

———. 2009. *Sustainable Energy—Without the Hot Air*. Cambridge: UIT Cambridge.

Mackinlay, J. D. 1986. Automating the design of graphical presentations of relational information. *ACM Transactions on Graphics* 5: 110–141.

McDaniel, E., and S. McDaniel. 2012a. *The Accidental Analyst: Show Your Data Who's Boss*. Seattle, WA: Freakalytics.

McDaniel, S., and E. McDaniel. 2012b. *Rapid Graphs with Tableau Software 7: Create Intuitive, Actionable Insights in Just 15 Days*. Seattle, WA: Freakalytics.

Morrison, P. S. 1985. Symbolic representation of tabular data. *New Zealand Journal of Geography* 79: 11–18.

Murphy, K. P. 2012. *Machine Learning: A Probabilistic Perspective*. Cambridge, MA: MIT Press.

Pirolli, P., and R. Rao. 1996. Table lens as a tool for making sense of data. In *AVI '96: Proceedings of the Workshop on Advanced Visual Interfaces*, ed. T. Catarci, M. F. Costabile, S. Levialdi, and G. Santucci, 67–80. New York: Association for Computing Machinery.

Playfair, W. H. 1786. *The Commercial and Political Atlas*. London: Robinson, Sewell, and Debrett.

———. 2005. *The Commercial and Political Atlas and Statistical Breviary*. Cambridge University Press: Cambridge. Edited by H. Wainer and I. Spence.

Rao, R., and S. K. Card. 1994. The table lens: Merging graphical and symbolic representations in an interactive focus + context visualization for tabular information. In *CHI '94: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ed. B. Adelson, S. Dumais, and J. S. Olson, 318–322. New York: Association for Computing Machinery.

Roberts, D. H., D. J. A. Evans, J. Lodwick, and N. J. Cox. 2013. The subglacial and ice-marginal signature of the North Sea Lobe of the British–Irish ice sheet during the last glacial maximum at Upgang, North Yorkshire, UK. *Proceedings of the Geologists' Association* 124: 503–519.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams. 1986. Learning representations by back-propagating errors. *Nature* 323: 533–536.

Sears, P. B. 1933. Climatic change as a factor in forest succession. *Journal of Forestry* 31: 934–942.

———. 1935. Types of North American pollen profiles. *Ecology* 16: 488–499.

Spence, R. 2007. *Information Visualization: Design for Interaction*. 2nd ed. Harlow, UK: Pearson.

Theus, M., and S. Urbanek. 2009. *Interactive Graphics for Data Analysis: Principles and Examples*. Boca Raton, FL: Chapman & Hall/CRC.

Unwin, A. 2015. *Graphical Data Analysis with R*. Boca Raton, FL: Taylor & Francis.

Unwin, A., M. Theus, and H. Hofmann. 2006. *Graphics of Large Datasets: Visualizing a Million*. New York: Springer.

Wainer, H. 2005. *Graphic Discovery: A Trout in the Milk and Other Visual Adventures*. Princeton, NJ: Princeton University Press.

———. 2009. *Picturing the Uncertain World: How to Understand, Communicate, and Control Uncertainty through Graphical Display*. Princeton, NJ: Princeton University Press.

Ward, M., G. Grinstein, and D. Keim. 2010. *Interactive Data Visualization: Foundations, Techniques, and Applications*. Natick, MA: A K Peters.

About the author

Nicholas Cox is a statistically minded geographer at Durham University. He contributes talks, postings, FAQs, and programs to the Stata user community. He has also coauthored 15 commands in official Stata. He was an author of several inserts in the *Stata Technical Bulletin* and is an editor of the *Stata Journal*. His "Speaking Stata" articles on graphics from 2004 to 2013 have been collected as *Speaking Stata Graphics* (College Station, TX: Stata Press, 2014).