The Stata Journal (2023)
23, Number 4, pp. 1086–1095 DOI: 10.1177/1536867X231212453

Speaking Stata: Finding the denominator: Minimum sample size from percentages

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

Abstract. Percentage breakdowns for a series of classes or categories are sometimes reported without a specification of class frequencies or even the total sample size. This column surveys the problem of estimating the minimum sample size and class frequencies consistent with a reported breakdown and a particular resolution. I introduce and explain a new command, `find_denom`. Rounding quirks whereby a total is reported as above or below 100% are discussed as a complication.

Keywords: st0737, find_denom, sample size, percentages, resolution, rounding, chi-squared tests, data checking, data quality

1 Introduction

An old joke with many variants has the following flavor: A naïve researcher is reporting on a project in which 33% of the sample said A and 33% said B, but the other person refused to answer. It is immediate that the sample size was 3. However, there is a more challenging twist: What denominator or sample size underlies a percentage breakdown of 40, 40, and 20? That breakdown is consistent with a sample size of 5, with 2, 2, and 1 as class frequencies. It is also consistent with any multiple of 5 and, dependent on the amount of rounding, reportably consistent with many other percentage breakdowns too. Thus, 2001, 1999, and 1000 yields exactly 40.02, 39.98, and 20.00 as a percentage breakdown and so rounds to 40.0, 40.0, and 20.0 when rounding to one decimal place. So would 2002, 1998, and 1000, and so would many other possibilities.

Every researcher should know that sample size should always be reported. Every researcher with any experience knows that does not always happen, and the culprits are not confined to advertising, journalism, or politics. Beyond hinting at possible ethical issues, this column concentrates on the technicalities of trying to guess the minimum sample size consistent with a reported percentage breakdown. We assume honest and accurate reporting, other than the sample size being suppressed or at least omitted.

The column introduces some basic tricks for calculating minimum sample sizes together with a command, `find_denom`. Section 2 gives some examples and a general discussion of the problem. Section 3 shows how `find_denom` may be used in practice and introduces further twists, especially the need to consider rounding quirks. Section 4 gives a more formal statement of the syntax of `find_denom`.

2 The problem: Examples and previous work

The problem was discussed by Wallis and Roberts (1956, 185–189) and in much more technical detail by Becker, Chambers, and Wilks (1988, 272–277). Two ideas arise immediately. First, a complete set of percentages is not needed to say something about minimum sample size. Thus, one percentage reported as 33% implies that the sample size cannot be 2 and must be at least 3. Second, the smallest percentage reported, or, if it is smaller, the smallest positive difference between two percentages reported, gives another handle on the minimum sample size. Thus, with a percentage breakdown of 40, 30, and 30, the smallest positive difference is 10, and equivalently 100/10 = 10 is the minimum sample size.

Wallis and Roberts (1956, 186) reported a fictitious percentage breakdown:

```
23.1
15.4
30.8
19.2
 7.7
 3.8
```

From that, both the smallest percentage and the smallest positive difference are 3.8, suggesting a minimum sample size of 100/3.8, which rounds as an integer to 26. The implied frequencies are thus

```
6
4
8
5
2
1
```

Wallis and Roberts (1956, 187–188) also reported percentage breakdowns of movie ratings from *Consumer Reports* August 1949, page 383. In turn, the categories are percentages reporting Excellent, Good, Fair, and Poor. Some examples are

Alias Nick Beal	6	27	47	20
Bride of Vengeance	11	22	56	11

Becker, Chambers, and Wilks (1988, 272) reported these percentages for considering vendors for 1986 from a personal computer magazine:

Ours	14.6
A	12.2
B	12.2
C	7.3
D	7.3

They gave an algorithm and S code for input expressed as proportions. The idea is just to bump up the sample size until implied percentages are all consistent with the stated results. It is this algorithm, translated from S to Stata but adapted for percentage input, that is implemented in the `find_denom` command.

Becker, Chambers, and Wilks (1988, 274–277) further discussed speeding up computations and allowing a certain number of outliers, essentially percentages that do not fit, say, because they were reported incorrectly. These elaborations are not implemented here but should be of interest for any deeper study.

Using a pie chart, Utts and Heckard (2022, 22) reported percentages from a group of students of answers 1(1)10 given the instruction "Randomly pick a number between 1 and 10". The percentages were 1.1 (for 1), 4.7, 11.6, 11.1, 9.5, 12.1, 29.5, 10.0, 7.4, and 3.2 (for 10). The same chart is also given in another text by Utts and Heckard (2006, 21).

Random digit choice is perhaps the most arresting of these examples, mostly for other reasons. If people were good random-number generators, then for the reference distribution of a discrete uniform (rectangular, flat) distribution, we would expect nearly equal percentages of around 10% each or, equivalently, probabilities of around 0.1. I prefer a bar chart to a pie chart, and that will come later (figure 1). If you are engaged in teaching, you may wish to use this example as a salutary warning of the deficiencies of people as random-number generators or as an intriguing illustration of the vagaries of number preferences.

3 Introducing find_denom and further twists

3.1 The idea of find_denom

`find_denom` is intended primarily for interactive use. You start with one or more percentages and want to know the minimum sample size consistent with those percentages. You must specify a tolerance through the option `epsilon()`. If the option name seems cryptic, there are two motivations. The first is the mathematical use of epsilon ϵ since Cauchy and Weierstrass to indicate small changes in an argument within rigorous analytic proofs. The second is the use by the mathematician Paul (Pál) Erdős, who certainly knew the first meaning, of epsilons to mean children, namely, people in small sizes (Hoffman 1998; Schechter 1998).

3.2 The idea of resolution

Naming the option `epsilon()`, a variant on the name used by Becker, Chambers, and Wilks (1988) in their S code, sidesteps a small issue of terminology. How do we refer to the difference in presentation produced by rounding to different particular powers of 10, say, the nearest multiple of $\ldots, 10, 1, 0.1, 0.01, \ldots$? Talking about the "number of decimal places" fits whenever the numbers are fractions, but not otherwise. The expression "number of significant figures" similarly only sometimes works well. Talking about "precision" runs into a problem where many statistical people want to use that term to refer to the variability (reliability) of repeated measurements, a widespread if not universal usage (see, for example, Eisenhart [1968]). I commend the term "resolution" broadly as used by Murphy (1997) and Crowder et al. (2020). This term has the secondary advantage that it also covers cases where the resolution is a matter of rounding to halves, quarters, eighths, other fractions that are powers of 2, or even to yet more unusual fractions. That said, such cases are not further discussed here.

3.3 Supposedly random digits

Let us now focus on the data example of supposedly random digits. We just type the reported percentages on the command line, but we must specify the option `epsilon()`, which can be abbreviated `eps()`, to be interpreted this way: `eps(0.05)` implies, for example, that 1.1, if known in more detail, would be somewhere between 1.05 and almost 1.15. The argument fed to `epsilon()` is thus half the resolution.

```
. find_denom 1.1 4.7 11.6 11.1 9.5 12.1 29.5 10.0 7.4 3.2, epsilon(0.05)
```

The command loops through possible sample sizes until it finds the smallest size consistent with all the information.

```
total of percentages is 100.2
minimum sample size is 190
minimum frequencies are 2 9 22 21 18 23 56 19 14 6
```

Immediately, we see that the total of the percentages is not exactly 100.0%. We will come back to this puzzle shortly.

The results match an accompanying histogram and most crucially the frequencies later listed by Utts and Heckard (2006, 241; 2022, 271). Here is a bar chart (figure 1).

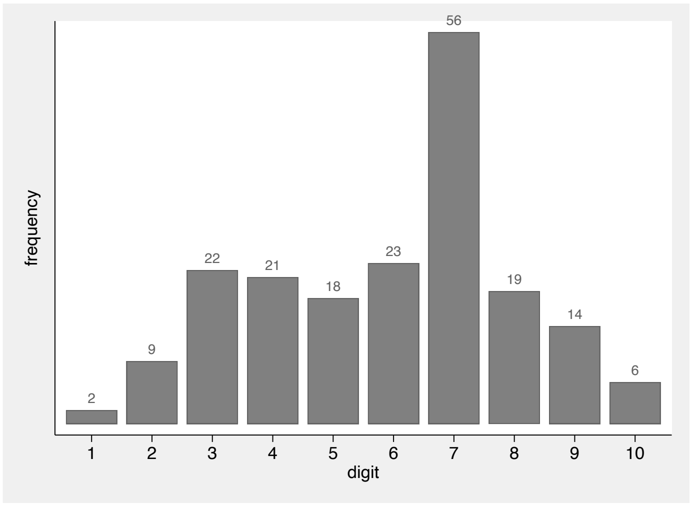

Figure 1. Frequencies of digit choice in data cited by Utts and Heckard in various texts

Incidentally, we can now get a chi-squared test or any other desired test that depends on knowing exact frequencies. One way to get a chi-squared test directly is to treat Mata as a convenient calculator. A null hypothesis of uniform distribution implies that each digit between 0 and 9 has an expected frequency of 190/10 = 19. The chi-squared statistic is thus, for observed frequencies, f_j:

$$\sum_{j=1}^{10} \frac{(f_j - 19)^2}{19} =: X^2$$

Here the notation X^2 (following, for example, Bishop, Fienberg, and Holland [1975]; Agresti [2013]) matches a strong convention to reserve Greek letters for population or theoretical quantities and to use Roman letters for sample statistics.

The number of degrees of freedom is the number of cells, 10, minus 1. The p-value is minute, which should seem unsurprising with such a marked departure from uniformity.

```
. mata
─────────────────────────────────────────────── mata (type end to exit) ──────
: observed = (2, 9, 22, 21, 18, 23, 56, 19, 14, 6)

: expected = rowsum(observed) / 10

: chisq = rowsum((observed :- expected):^2 :/ expected)

: chisq
  104.3157895

: chi2tail(9, chisq)
  2.10204e-18

: end
──────────────────────────────────────────────────────────────────────────────
```

If you are new to Mata, the least predictable detail here is that `:-`, `:^2`, and `:/` are syntax for various elementwise operations: elementwise subtraction, squaring, and division, respectively.

Oddly, or otherwise, there is no official command in Stata for this test for a one-way table. However, `chitesti` from the community-contributed package `tab_chi` (Cox 1999) on the Statistical Software Components Archive is available.

3.4 Taking account of rounding

As already implied, `find_denom` has really only one idea. Essentially, it is optimistic that reported percentages are correct and consistent. However, we already have run into a small but fundamental problem. Rounded percentages need not sum to exactly 100, even with careful and consistent calculation. Our detailed example discussed just now showed that straight away. The phenomenon is familiar to experienced researchers. For incisive discussion and analysis, see Mosteller, Youtz, and Zahn (1967) and Diaconis and Freedman (1979).

We can look at the fine structure of our example again using Mata. We transpose the vector of observed counts for convenience because a table with more rows than columns is easier to work with than its transpose. We align the column of observed counts with percentage displays for resolutions 1, 0.1, and 0.01.

```
: observed = (2, 9, 22, 21, 18, 23, 56, 19, 14, 6)
: observed' , round(100 * observed' :/ 190, (0.1, 0.01, 0.001))
            1        2        3        4
    ┌─────────────────────────────────────┐
  1 │       2      1.1     1.05    1.053  │
  2 │       9      4.7     4.74    4.737  │
  3 │      22     11.6    11.58   11.579  │
  4 │      21     11.1    11.05   11.053  │
  5 │      18      9.5     9.47    9.474  │
  6 │      23     12.1    12.11   12.105  │
  7 │      56     29.5    29.47   29.474  │
  8 │      19       10       10       10  │
  9 │      14      7.4     7.37    7.368  │
 10 │       6      3.2     3.16    3.158  │
    └─────────────────────────────────────┘

: colsum(round(100 * observed' :/ 190, (0.1, 0.01, 0.001)))
            1          2          3
   ┌────────────────────────────────┐
 1 │    100.2        100   100.001  │
   └────────────────────────────────┘
```

There is nothing untoward here. It just happened that with a resolution of 0.1, there was more rounding up than rounding down. With a resolution of 0.01, the total would have appeared exactly correct, but with a resolution of 0.001, the total would have appeared off by 0.001.

In general, you should check that the total looks plausible, but note that—as in this example—the command does not throw you out if the total is not exactly 100%. Not only could it easily fall above or below as a matter of rounding quirks, but there also could be precision problems from holding decimals using binary representations. If the latter are unfamiliar to you, type `search precision` for pointers to many resources. I particularly recommend blog posts on precision by William Gould.

For more on rounding and its cousin binning in Stata, see Cox (2018).

3.5 Further cautionary notes

Hence, although `find_denom` is offered as a tool that may be useful, you need to be on the lookout for rounding quirks. Let me add some further cautions:

1. The user can and should flag that percentages are partial or incomplete if they are. The option `partial` is intended for this purpose.
2. On occasion, the algorithm converges on an inconsistent solution, in which case there will be a report to that effect.
3. The `dots` option is provided to show progress and may indirectly indicate an insoluble problem.

3.6 Other examples

If interested, you can run the command for the other examples given previously. Note the simple but crucial detail that `epsilon()` will vary according to the resolution of the data.

```
. find_denom 23.1 15.4 30.8 19.2 7.7 3.8, epsilon(0.05)
. find_denom 6 27 47 20, epsilon(0.5)
. find_denom 11 22 56 11, epsilon(0.5)
. find_denom 14.6 12.2 12.2 7.3 7.3, epsilon(0.05)
```

4 The find_denom command

4.1 Syntax

`find_denom` #1 [#2 ...], `eps`<u>ilon(#)</u> [`par`tial `dots`]

4.2 Description

`find_denom` reports minimum sample size and minimum frequencies given one or more percentages rounded to some precision or resolution.

4.3 Options

`epsilon(#)` is a required option indicating half the perceived resolution. Thus, if percentages are rounded to integers, specify `epsilon(0.5)`; if they are rounded to 1 decimal place, specify `epsilon(0.05)`. The thinking is that a report of # means that the true value is between # − epsilon and (almost) # + epsilon.

`partial` is a flag that you know that the specified percentage or percentages are partial or incomplete. For example, `find_denom 33.3, eps(0.05) partial` is allowed (and required) as a test that will show one percentage of 33.3 to be consistent with a class frequency of 1 and a total frequency of 3.

`dots` calls up minor entertainment in the form of a dot or period for every sample size tried and a printout of the sample size at every multiple of 50. This option may also be useful for showing that a solution cannot be found.

5 Conclusion

Suppression or omission of percentages in a report can be troublesome, whether it was just careless or even raises questions of malpractice. Whatever the circumstances, `find_denom` is offered as a small and simple command for analysis.

A point of likely interest to programmers is that I first translated from another language (S) to Mata and then wrote the command as a wrapper around the Mata code. More generally, examples here are intended to underline the utility of Mata as an online calculator and display tool.

6 Programs and supplemental materials

To install a snapshot of the corresponding software files as they existed at the time of publication of this article, type

```
. net sj 23-4
. net install st0737      (to install program files, if available)
. net get st0737          (to install ancillary files, if available)
```

7 References

Agresti, A. 2013. *Categorical Data Analysis*. 3rd ed. Hoboken, NJ: Wiley.

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.

Bishop, Y. M. M., S. E. Fienberg, and P. W. Holland. 1975. *Discrete Multivariate Analysis: Theory and Practice*. Cambridge, MA: MIT Press. Reissue, New York: Springer, 2007.

Cox, N. J. 1999. tab_chi: Stata modules for tabulation of multiple variables in Stata 8.2 or better. Statistical Software Components S368901, Department of Economics, Boston College. https://ideas.repec.org/c/boc/bocode/s368901.html.

———. 2018. Speaking Stata: From rounding to binning. *Stata Journal* 18: 741–754. https://doi.org/10.1177/1536867X1801800311.

Crowder, S., C. Delker, E. Forrest, and N. Martin. 2020. *Introduction to Statistics in Metrology*. Cham, Switzerland: Springer. https://doi.org/10.1007/978-3-030-53329-8.

Diaconis, P., and D. Freedman. 1979. On rounding percentages. *Journal of the American Statistical Association* 74: 359–364. https://doi.org/10.2307/2286335.

Eisenhart, C. 1968. Expression of the uncertainties of final results. *Science* 160: 1201–1204. https://doi.org/10.1126/science.160.3833.1201.

Hoffman, P. 1998. *The Man Who Loved Only Numbers: The Story of Paul Erdős and the Search for Mathematical Truth*. London: Fourth Estate.

Mosteller, F., C. Youtz, and D. Zahn. 1967. The distribution of sums of rounded percentages. *Demography* 4: 850–858. https://doi.org/10.2307/2060324.

Murphy, E. A. 1997. *The Logic of Medicine*. 2nd ed. Baltimore: Johns Hopkins University Press.

Schechter, B. 1998. *My Brain is Open: The Mathematical Journeys of Paul Erdős*. New York: Simon and Schuster.

Utts, J. M., and R. F. Heckard. 2006. *Statistical Ideas and Methods*. Belmont, CA: Thomson Brooks/Cole.

———. 2022. *Mind on Statistics*. 6th ed. Boston: Cengage Learning.

Wallis, W. A., and H. V. Roberts. 1956. *Statistics: A New Approach*. Glencoe, IL: Free Press.

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 16 commands in official Stata. He was an author of several inserts in the *Stata Technical Bulletin* and is Editor-at-Large of the *Stata Journal*. His "Speaking Stata" articles on graphics from 2004 to 2013 have been collected as *Speaking Stata Graphics* (2014, College Station, TX: Stata Press).