The Stata Journal

The *Stata Journal* publishes reviewed papers together with shorter notes or comments, regular columns, book reviews, and other material of interest to Stata users. Examples of the types of papers include 1) expository papers that link the use of Stata commands or programs to associated principles, such as those that will serve as tutorials for users first encountering a new field of statistics or a major new technique; 2) papers that go "beyond the Stata manual" in explaining key features or uses of Stata that are of interest to intermediate or advanced users of Stata; 3) papers that discuss new commands or Stata programs of interest either to a wide spectrum of users (e.g., in data management or graphics) or to some large segment of Stata users (e.g., in survey statistics, survival analysis, panel analysis, or limited dependent variable modeling); 4) papers analyzing the statistical properties of new or existing estimators and tests in Stata; 5) papers that could be of interest or usefulness to researchers, especially in fields that are of practical importance but are not often included in texts or other journals, such as the use of Stata in managing datasets, especially large datasets, with advice from hard-won experience; and 6) papers of interest to those who teach, including Stata with topics such as extended examples of techniques and interpretation of results, simulations of statistical concepts, and overviews of subject areas.

For more information on the *Stata Journal*, including information for authors, see the webpage

http://www.stata-journal.com

The *Stata Journal* is indexed and abstracted in the following:

- CompuMath Citation Index®
- Current Contents/Social and Behavioral Sciences®
- RePEc: Research Papers in Economics
- Science Citation Index Expanded (also known as SciSearch®)
- Scopus™
- Social Sciences Citation Index®

The *Stata Journal*, electronic version (ISSN 1536-8734) is a publication of Stata Press. Stata, Mata, NetCourse, and Stata Press are registered trademarks of StataCorp LP.

The Stata Journal (2012)
12, Number 1, pp. 159–161

Stata tip 105: Daily dates with missing days

Steven J. Samuels
18 Cantine's Island
Saugerties, NY 12477
USA
sjsamuels@gmail.com

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

1 Introduction

In projects that record daily dates, there may be some observations for which only the month and year are known. This can happen, for example, when memories of events are fuzzy or written records were never kept or have been lost.

In this tip, we suggest some simple strategies for dealing with missing daily dates. Whatever is done should, naturally, be done cautiously. In addition to solving a real problem met in data handling, the strategies we suggest here provide a good exercise in combining Stata date functions. For the sake of clarity, we work with more new variables than is strictly necessary. Toward the end of the tip, we will explain how one new variable should be kept as a record of what was done.

The process of handling complete dates is easy. If the year, month, and day are recorded as separate variables—say, `year`, `month`, and `day`—we can

```
. generate dailydate = mdy(month,day,year)
```

If the date is imported in a string variable—say, `reported`—either delimited (as in `"2008/1/13"`) or run together (as in `"20080113"`), we can use Stata's powerful date parser:

```
. generate dailydate = date(reported, "YMD")
```

These `generate` commands will return missing dates if the day is missing. We still want to use the information we have for month and year. If we have read in a string variable, we can create `year` and `month` numeric variables to hold the known year and month. For example, if the date is run together as above, then

```
. generate month = substr(reported,1,4)
. generate year = substr(reported,5,.)
```

This puts the two forms of input (separate numeric variables and strings) on equal footing.

2 If a monthly date is enough

In some situations, the analyst might decide that knowing the month of the event is sufficient. For example, the event might be a medical test that should be performed every month. If so, we can create a monthly date with Stata's `monthly()` function:

```
. generate int monthlydate = monthly(year + "-" + month, "YM")
```

For `"200801"`, this recipe yields 576 as a monthly date (January 1960 is month 0). If `monthlydate` is given a `%tm` format, it will display as `"2008m1"`.

3 Use a midpoint date

If months are not enough, then a common work-around is to substitute 15 for the missing day. In a nonleap year, the mean number of days in each month is

$$(4/12) \times 30 + (7/12) \times 31 + (1/12) \times 28 = 30.416667$$

In a leap year, it is 30.5. Therefore, on average, 15 is the integer closest to the mean. With this choice, we can be assured that the error in each date is no more than ± 16 days in any month.

```
. generate imputedday = 15
```

Substitution of 15 for the missing day is an example of *mean imputation* (Little and Rubin 2002; Seastrom, Kaufman, and Lee 2002). In general, mean imputation is undesirable because it produces spikes at the mean and distorts distributional parameters such as the variance. This undesirable behavior can also carry over to the distribution of intervals between imputed dates. For example, the absolute difference of two dates, one imputed to be the 15th, in one month will have a range no more than 16 days, compared with 30 or 31 days for the original data.

We can prevent this distortion by randomly imputing the day. A simple procedure is to choose a day at random from the 28, 29, 30, or 31 days of the month with the missing day. But exactly what is the correct number of days? This tip was first drafted 10 October 2011. For most people, 31 is an easy answer for the correct number of days in October. It is easy for the authors because early in their educations, they memorized a little poem about the months that included the exceptions for February and leap years.

But how do we get Stata to do this calculation? What is crucial here is that Stata already knows the calendar, so it does not need to be instructed about month lengths or leap years. Instead, the trick is to recognize that the last day of the current month—in this example, 31—is just one day before the first day of the next month. Thus we can get this by calculating

```
. generate modays = day(dofm(monthlydate + 1) - 1)
```

`monthlydate + 1` is the next month; `dofm()` yields the first day of that month as a daily date and subtracting 1 gives us the last day of this month as a daily date; finally, `day()` yields the day of the month of that last day. This works regardless of leap years or whether the next month is also in the next year.

Then we can randomly impute a day in the month by calculating

```
. generate imputedday = ceil(modays * runiform())
```

However we impute the date, we can now fix the missing dates:

```
. replace dailydate = date(reported + string(imputedday), "YMD")
> if missing(dailydate)
```

We leave behind the variable `imputedday` in the dataset, because its nonmissing values indicate what was imputed and where each was imputed in the data.

The assumption entailed by random sampling from a discrete uniform distribution is clearly that any day of the month has the same probability. If that seems unlikely—for example, because of seasonality—some other procedure may be advisable. In the same vein, it will be recognized that the imputation does not use any other information that might be available. In the original medical example, that could include other observations on the same patient. Although these and other complications may be problematic in practice, we leave them for another day.

References

Little, R. J. A., and D. B. Rubin. 2002. *Statistical Analysis with Missing Data*. 2nd ed. Hoboken, NJ: Wiley.

Seastrom, M. M., S. Kaufman, and R. Lee. 2002. Appendix B: Evaluating the impact of imputations for item nonresponse. In *NCES Statistical Standards (NCES 2003601)*, ed. M. M. Seastrom. National Center for Education Statistics, Institute of Education Sciences. http://nces.ed.gov/statprog/2002/appendixb.asp.