How can I “fill down” or expand observations with respect to a time variable in Stata?

Authored by

stats writer

July 1, 2024

The process of expanding or "filling down" observations with respect to a time variable in Stata involves duplicating data points for each time point within a specified time range. This allows for a more comprehensive analysis of time series data, as it ensures that all time points are accounted for in the dataset. This can be accomplished by using the "expand" command in Stata, which allows users to specify the desired time range and create new observations for each time point. This feature is useful for performing various statistical analyses, such as panel data analysis or trend analysis, and can provide more accurate and complete results.

How can I "fill down"/expand observations with respect to a time variable? | Stata FAQ

A time series data set may have gaps and sometimes we may want to fill in the gaps so the time variable will be in consecutive order. This involves two steps. First of all, we need to expand the data set so the time variable is in the right form. When we expand the data, we will inevitably create missing values for other variables. The second step is to replace the missing values sensibly. The examples shown here use Stata's command tsfill and a user-written command "carryforward" by David Kantor to perform the two steps described above. You can download the "carryforward" via "search carryforward" in

**Stata (see How can I
use the search command to search for programs and get additional help? for
more information about using search) and following the appropriate link.
What the command carryforward does is to carry values forward from one
observation to the next, filling in missing values with the previous value.**

Example 1, a simple fill with carryforward

**In this example, the starting and end point could be different for different
individuals and the gaps are filled in by individuals.**

```
clear
input id time y
1 1 1.2
1 3 2.4
1 4 3.4
1 7 3.2
1 9 2.4
2 3 1.8
2 4 5.6
```

2 6 4.3
3 2 2.3
3 4 4.5
3 7 6.7
end

tsset id time
panel variable: id (unbalanced)
time variable: time, 1 to 9, but with gaps
delta: 1 unit

tsfill
list, clean noobs

id time y
1 1 1.2
1 2 .
1 3 2.4
1 4 3.4
1 5 .
1 6 .
1 7 3.2
1 8 .
1 9 2.4

2 3 1.8

2 4 5.6

2 5 .

2 6 4.3

3 2 2.3

3 3 .

3 4 4.5

3 5 .

3 6 .

3 7 6.7

bysort id: carryforward y, gen(yn)

(8 real changes made)

list, clean noobs

id time y yn

1 1 1.2 1.2

1 2 . 1.2

1 3 2.4 2.4

1 4 3.4 3.4

1 5 . 3.4

1 6 . 3.4

1 7 3.2 3.2

1 8 . 3.2

1 9 2.4 2.4

2 3 1.8 1.8

2 4 5.6 5.6

2 5 . 5.6

2 6 4.3 4.3

3 2 2.3 2.3

3 3 . 2.3

3 4 4.5 4.5

3 5 . 4.5

3 6 . 4.5

3 7 6.7 6.7

Example 2

Sometimes, we might want to get a completely balanced data. In this case, the
starting point will be the same for all the individuals and the end point will
be the same for all the individuals as well. To this end, the option "full"
for tsfill is used.

```
clear
input id time y
```

1 1 1.2
1 3 2.4
1 4 3.4
1 7 3.2
1 9 2.4
2 3 1.8
2 4 5.6
2 6 4.3
3 2 2.3
3 4 4.5
3 7 6.7
end

tsset id time
panel variable: id (unbalanced)
time variable: time, 1 to 9, but with gaps
delta: 1 unit

tsfill, full

list, clean noobs

id time y
1 1 1.2
1 2 .

1 3 2.4

1 4 3.4

1 5 .

1 6 .

1 7 3.2

1 8 .

1 9 2.4

2 1 .

2 2 .

2 3 1.8

2 4 5.6

2 5 .

2 6 4.3

2 7 .

2 8 .

2 9 .

3 1 .

3 2 2.3

3 3 .

3 4 4.5

3 5 .

3 6 .

3 7 6.7

3 8 .

3 9 .

bysort id: carryforward y, gen(yn)
(13 real changes made)

list, clean noobs

id time y yn
1 1 1.2 1.2
1 2 . 1.2
1 3 2.4 2.4
1 4 3.4 3.4
1 5 . 3.4
1 6 . 3.4
1 7 3.2 3.2
1 8 . 3.2
1 9 2.4 2.4
2 1 . .
2 2 . .
2 3 1.8 1.8
2 4 5.6 5.6
2 5 . 5.6
2 6 4.3 4.3
2 7 . 4.3

2 8 . 4.3

2 9 . 4.3

3 1 . .

3 2 2.3 2.3

3 3 . 2.3

3 4 4.5 4.5

3 5 . 4.5

3 6 . 4.5

3 7 6.7 6.7

3 8 . 6.7

3 9 . 6.7

Example 3

In previous example, we see that not all the missing values are replaced
since "carryforward" does not carry backforward. We can replace the
missing values by performing one more "carryforward" in a backward way.
Although this is possible to do, it does not mean that it is a good idea to do.
This example is merely for the purpose of illustration.

clear

```
input id time y
1 1 1.2
1 3 2.4
1 4 3.4
1 7 3.2
1 9 2.4
2 3 1.8
2 4 5.6
2 6 4.3
3 2 2.3
3 4 4.5
3 7 6.7
end

tsset id time
tsfill, full
bysort id: carryforward y, gen(yn)

gsort id - time
bysort id: carryforward yn, gen(yfinal)

list, clean noobs

id time y yn yfinal
```

1 9 2.4 2.4 2.4

1 8 . 3.2 3.2

1 7 3.2 3.2 3.2

1 6 . 3.4 3.4

1 5 . 3.4 3.4

1 4 3.4 3.4 3.4

1 3 2.4 2.4 2.4

1 2 . 1.2 1.2

1 1 1.2 1.2 1.2

2 9 . 4.3 4.3

2 8 . 4.3 4.3

2 7 . 4.3 4.3

2 6 4.3 4.3 4.3

2 5 . 5.6 5.6

2 4 5.6 5.6 5.6

2 3 1.8 1.8 1.8

2 2 . . 1.8

2 1 . . 1.8

3 9 . 6.7 6.7

3 8 . 6.7 6.7

3 7 6.7 6.7 6.7

3 6 . 4.5 4.5

3 5 . 4.5 4.5

3 4 4.5 4.5 4.5

3 3 . 2.3 2.3

3 2 2.3 2.3 2.3

3 1 . . 2.3