tsappend — Add observations to a time-series dataset

Syntax

```stata
tsappend , { add(#) | last(date | clock) tsfmt(string) } [ options ]
```

<table>
<thead>
<tr>
<th>options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>* add(#)</td>
<td>add # observations</td>
</tr>
<tr>
<td>* last(date</td>
<td>clock)</td>
</tr>
<tr>
<td>* tsfmt(string)</td>
<td>use time-series function string with last(date</td>
</tr>
<tr>
<td>panel(panel_id)</td>
<td>add observations to panel panel_id</td>
</tr>
</tbody>
</table>

* Either add(#) is required, or last(date | clock) and tsfmt(string) are required.

You must tsset your data before using tsappend; see [TS] tsset.

Menu

Statistics > Time series > Setup and utilities > Add observations to time-series dataset

Description

tappend appends observations to a time-series dataset or to a panel dataset. tsappend uses and updates the information set by tsset.

Options

add(#) specifies the number of observations to add.

last(date | clock) and tsfmt(string) must be specified together and are an alternative to add().

last(date | clock) specifies the date or the date and time of the last observation to add.

tsfmt(string) specifies the name of the Stata time-series function to use in converting the date specified in last() to an integer. The function names are tc (clock), tC (Clock), td (daily), tw (weekly), tm (monthly), tq (quarterly), and th (half-yearly).

For clock times, the last time added (if any) will be earlier than the time requested in last(date | clock) if last() is not a multiple of delta units from the last time in the data.

For instance, you might specify last(17may2007) tsfmt(td), last(2001m1) tsfmt(tm), or last(17may2007 15:30:00) tsfmt(tc).

panel(panel_id) specifies that observations be added only to panels with the ID specified in panel().
Remarks and examples

Remarks are presented under the following headings:

Introduction
Using tsappend with time-series data
Using tsappend with panel data

Introduction

*tsappend* adds observations to a time-series dataset or to a panel dataset. You must *tsset* your data before using *tsappend*. *tsappend* simultaneously removes any gaps from the dataset.

There are two ways to use *tsappend*: you can specify the *add(#)* option to request that # observations be added, or you can specify the *last(date|clock)* option to request that observations be appended until the date specified is reached. If you specify *last()* you must also specify *tsfmt()*.

*tsfmt()* specifies the Stata time-series date function that converts the date held in *last()* to an integer.

*tsappend* works with time series of panel data. With panel data, *tsappend* adds the requested observations to all the panels, unless the *panel()* option is also specified.

Using tsappend with time-series data

*tsappend* can be useful for appending observations when dynamically predicting a time series. Consider an example in which *tsappend* adds the extra observations before dynamically predicting from an AR(1) regression:

```
use http://www.stata-press.com/data/r13/tsappend1
regress y l.y
```

```
Source | SS       df   MS
--------|----------|---------|---------|---------|
Model   | 115.349555 1  115.349555 Prob > F = 0.0000
Residual | 461.241577 477 .966963473 R-squared = 0.2001
        | 576.591132 478 1.2062576 Root MSE = .98334
```

```
y | Coef. Std. Err. t    P>|t|    [95% Conf. Interval]
L1 | .4493507 .0411417 10.92 0.000  .3685093   .5301921
_cons | 11.11877 .8314581 13.37 0.000  9.484993   12.75254
```

```
matrix b = e(b)
matrix colnames b = L.xb one
.tsset
time variable: t2, 1960m2 to 2000m1
delta: 1 month
.tsappend, add(12)
.tsset
time variable: t2, 1960m2 to 2001m1
delta: 1 month
.predict xb if t2<=tm(2000m2)
(option xb assumed; fitted values)
(12 missing values generated)
```
. gen one=1
. mat score xb=b if t2>=tm(2000m2), replace

The calls to `tsset` before and after `tsappend` were unnecessary. Their output reveals that `tsappend` added another year of observations. We then used `predict` and `matrix score` to obtain the dynamic predictions, which allows us to produce the following graph:

```
. line y xb t2 if t2>=tm(1995m1), ytitle("") xtitle("time")
```

![Graph showing dynamic predictions with time-series data](image)

In the call to `tsappend`, instead of saying that we wanted to add 12 observations, we could have specified that we wanted to fill in observations through the first month of 2001:

```
. use http://www.stata-press.com/data/r13/tsappend1, clear
. tsset
time variable: t2, 1960m2 to 2000m1
   delta: 1 month
. tsappend, last(2001m1) tsfmt(tm)
. tsset
time variable: t2, 1960m2 to 2001m1
   delta: 1 month
```

We specified the `tm()` function in the `tsfmt()` option. `[D] functions` contains a list of time-series functions for translating date literals to integers. Because we have monthly data, and since `[D] functions` tells us that we want to use the `tm()` function, we specified the `tsfmt(tm)` option.

The following table shows the most common types of time-series data, their formats, the appropriate translation functions, and the corresponding options for `tsappend`:

<table>
<thead>
<tr>
<th>Description</th>
<th>Format</th>
<th>Function</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>time</td>
<td>%tc</td>
<td>tc()</td>
<td>tsfmt(tc)</td>
</tr>
<tr>
<td>time</td>
<td>%tC</td>
<td>tC()</td>
<td>tsfmt(tC)</td>
</tr>
<tr>
<td>daily</td>
<td>%td</td>
<td>td()</td>
<td>tsfmt(td)</td>
</tr>
<tr>
<td>weekly</td>
<td>%tw</td>
<td>tw()</td>
<td>tsfmt(tw)</td>
</tr>
<tr>
<td>monthly</td>
<td>%tm</td>
<td>tm()</td>
<td>tsfmt(tm)</td>
</tr>
<tr>
<td>quarterly</td>
<td>%tq</td>
<td>tq()</td>
<td>tsfmt(tq)</td>
</tr>
<tr>
<td>half-yearly</td>
<td>%th</td>
<td>th()</td>
<td>tsfmt(th)</td>
</tr>
<tr>
<td>yearly</td>
<td>%ty</td>
<td>ty()</td>
<td>tsfmt(ty)</td>
</tr>
</tbody>
</table>
tsappend’s actions on panel data are similar to its action on time-series data, except that tsappend performs those actions on each time series within the panels.

If the end dates vary over panels, last() and add() will produce different results. add(#) always adds # observations to each panel. If the data end at different periods before tsappend, add() is used, the data will still end at different periods after tsappend, add(). In contrast, tsappend, last() tsfmt() will cause all the panels to end on the specified last date. If the beginning dates differ across panels, using tsappend, last() tsfmt() to provide a uniform ending date will not create balanced panels because the number of observations per panel will still differ.

Consider the panel data summarized in the output below:

```
. use http://www.stata-press.com/data/r13/tsappend3, clear
. xtdescribe
     id:  1, 2, ..., 3        n =       3
     t2: 1998m1, 1998m2, ..., 2000m1        T =      25
          Delta(t2) = 1 month
          Span(t2) = 25 periods
          (id*t2 uniquely identifies each observation)

     Distribution of T_i:  min 5%  25%  50%  75%  95% max
              13  13  13  20  24  24  24
     Freq. Percent       Cum. Pattern
          1 33.33       33.33
          1 33.33       66.67
          1 33.33      100.00

. by id: summarize t2
          -> id = 1
          Variable |     Obs  Mean   Std. Dev.     Min   Max
          ------- |-------|--------|--------|-------|-------
            t2 |     13 474.0 3.894444 468.0 480.0

          -> id = 2
          Variable |     Obs  Mean   Std. Dev.     Min   Max
          ------- |-------|--------|--------|-------|-------
            t2 |     20 465.5 5.91608 456.0 475.0

          -> id = 3
          Variable |     Obs  Mean   Std. Dev.     Min   Max
          ------- |-------|--------|--------|-------|-------
            t2 |     24 468.3333 7.322786 456.0 480.0
```

The output from xtdescribe and summarize on these data tells us that one panel starts later than the other, that another panel ends before the other two, and that the remaining panel has a gap in the time variable but otherwise spans the entire time frame.
Now consider the data after a call to tsappend, add(6):

```
.tsappend, add(6)
.xtdescribe

id: 1, 2, ..., 3  n = 3
T2: 1998m1, 1998m2, ..., 2000m7  T = 31
Delta(t2) = 1 month
Span(t2) = 31 periods
(id*t2 uniquely identifies each observation)
Distribution of T_i:  min  5%  25%  50%  75%  95%  max
19 19 19 26 31 31 31
Freq. Percent Cum. Pattern
1 33.33 33.33 ................1111111111111111111
1 33.33 66.67 1111111111111111111111111111111....
1 33.33 100.00 11111111111111111111111111111111111
3 100.00 Xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
.by id: summarize t2

-> id = 1
Variable |  Obs  |  Mean  |  Std. Dev.  |  Min  |  Max
---|---|---|---|---|---
t2 | 19 | 477 | 5.627314 | 468 | 486

-> id = 2
Variable |  Obs  |  Mean  |  Std. Dev.  |  Min  |  Max
---|---|---|---|---|---
t2 | 26 | 468.5 | 7.648529 | 456 | 481

-> id = 3
Variable |  Obs  |  Mean  |  Std. Dev.  |  Min  |  Max
---|---|---|---|---|---
t2 | 31 | 471 | 9.092121 | 456 | 486
```

This output from xtdescribe and summarize after the call to tsappend shows that the call to tsappend, add(6) added 6 observations to each panel and filled in the gap in the time variable in the second panel. tsappend, add() did not cause a uniform end date over the panels.

The following output illustrates the contrast between tsappend, add() and tsappend, last() tsfmt() with panel data that end at different dates. The output from xtdescribe and summarize shows that the call to tsappend, last() tsfmt() filled in the gap in t2 and caused all the panels to end at the specified end date. The output also shows that the panels remain unbalanced because one panel has a later entry date than the other two.
. use http://www.stata-press.com/data/r13/tsappend2, clear
. tsappend, last(2000m7) tsfmt(tm)
. xtdescribe

<table>
<thead>
<tr>
<th>id:</th>
<th>1, 2, ..., 3</th>
<th>n =</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>t2:</td>
<td>1998m1, 1998m2, ..., 2000m7</td>
<td>T =</td>
<td>31</td>
</tr>
<tr>
<td>Delta(t2) = 1 month</td>
<td>Span(t2) = 31 periods</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(id*t2 uniquely identifies each observation)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Distribution of T_{i}: min 5% 25% 50% 75% 95% max

<table>
<thead>
<tr>
<th>Freq.</th>
<th>Percent</th>
<th>Cum.</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>66.67</td>
<td>66.67</td>
</tr>
<tr>
<td>1</td>
<td>33.33</td>
<td>100.00</td>
</tr>
</tbody>
</table>

Pattern

11111111111111111111111111111111
............1111111111111111111

3 100.00

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

. by id: summarize t2

-> id = 1

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>t2</td>
<td>19</td>
<td>477</td>
<td>5.627314</td>
<td>468</td>
<td>486</td>
</tr>
</tbody>
</table>

-> id = 2

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>t2</td>
<td>31</td>
<td>471</td>
<td>9.092121</td>
<td>456</td>
<td>486</td>
</tr>
</tbody>
</table>

-> id = 3

<table>
<thead>
<tr>
<th>Variable</th>
<th>Obs</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>t2</td>
<td>31</td>
<td>471</td>
<td>9.092121</td>
<td>456</td>
<td>486</td>
</tr>
</tbody>
</table>

Stored results

tsappend stores the following in r():

Scalars

r(add) number of observations added

Also see

[TS] tsset — Declare data to be time-series data