A user guide for *panelstat*

Paulo Guimaraes, BPlim

28jun2018

1 Introduction

`panelstat` is a Stata user-written command to help understand the characteristics of standard panel data sets. Usage of `panelstat` command is quite simple:

`panelstat panelvar timevar [if] [in] [ , options]`

where `panelvar` is a unit identifier for the panel and `timevar` identifies the time variable. `panelstat` has many options which we will discuss using examples. For now, if you want to see the full list of options simply check the help file for `panelstat`.

2 Basic Usage

To illustrate the usage of `panelstat` we start by loading Stata's *nlswork* file set which is a sample of the National Longitudinal Survey of young women aged 14 to 26 years in 1968, and which were observed for several years.

```
. sysuse nlswork.dta, clear
(National Longitudinal Survey.  Young Women 14-26 years of age in 1968)
```

The unit identifier for this panel is *idcode* – a unique identifier for each participant in the study. The time variable is *year*. To obtain basic descriptive statistics for this panel we can run `panelstat`:

```
. panelstat idcode year

*****************************************************
Analyzing ./nlswork.dta
*****************************************************
*****************************************************
Basic descriptives
*****************************************************
There are 28534 time x individuals observations
There are 4711 unique individuals
Time values range from 68 to 88
```

```
Maximum time range is 21
The average number of periods per individual is 6.06
The level of completeness is 28.84% (100% is a fully balanced panel)
Average number of gaps per individual is 2.75
Average gap size is 1.84
Largest gap is 19
*****************************************************
*****************************************************
Distribution of number of observations per individual
*****************************************************
```

Observ per individual	Freq.	Percent	Cum.
1	547	11.61	11.61
2	498	10.57	22.18
3	484	10.27	32.46
4	411	8.72	41.18
5	421	8.94	50.12
6	398	8.45	58.57
7	345	7.32	65.89
8	323	6.86	72.74
9	302	6.41	79.16
10	270	5.73	84.89
11	202	4.29	89.17
12	158	3.35	92.53
13	147	3.12	95.65
14	119	2.53	98.17
15	86	1.83	100.00
Total	4,711	100.00	

```
*****************************************************
Number of individuals per time unit
*****************************************************
```

Time	Freq.	Percent	Cum.
68	1,375	4.82	4.82
69	1,232	4.32	9.14
70	1,686	5.91	15.05
71	1,851	6.49	21.53
72	1,693	5.93	27.47
73	1,981	6.94	34.41
75	2,141	7.50	41.91
77	2,171	7.61	49.52
78	1,964	6.88	56.40
80	1,847	6.47	62.88
82	2,085	7.31	70.18
83	1,987	6.96	77.15
85	2,085	7.31	84.45
87	2,164	7.58	92.04
88	2,272	7.96	100.00
Total	28,534	100.00	

As a result `panelstat` produced three pieces of information. The first block, under the heading *Basic Descriptives*, provides information on the total number of observations, individuals, and the time span of the panel. It also tries to give you an idea of how close the panel is to a fully balanced one. Since we have $4,711$ individuals and we observe data for a total of 21 years, if we had a fully balanced dataset we would have $4,711 \times 21 = 98,931$ observations. Since we only have $28,534$ observations the level of completeness is $28,534/98,931 = 28.84\%$.

The second block of information is simply a tabulation of the number of observations per individual. We can immediately see that there are 547 individuals with only one observation (these are called *singletons*) and that only 86 individuals are observed for 15 years - the largest number of periods an individual is observed.

The final block gives us a simple tabulation of the number observations in each year. To avoid calculation of the tables of "Basic Descriptives" we can use the `nosum` option.

3 General Options

The *excel* option

When working with `panelstat` you may want to save the output of your analyses to an excel spreadsheet. This is done by specifying the name of an output file where you want the results to be stored. For example,

```
panelstat idcode year, excel(myresults)
```

will create an excel spreadsheet that contains all information displayed by the command. By default, the new excel file will be created in Stata's working directory. You can, however, specify a full path as in

```
panelstat idcode year, excel("C:\mystuff\myresults")
```

If you use other `panelstat` options then all results will be stored in the same excel spreadsheet under different worksheets. To replace an existing spreadsheet you can specify the sub-option `replace` as in

```
panelstat idcode year, excel(myresults, replace)
```

and the sub-option `modify` will let you overwrite an existing spreadsheet.

```
panelstat idcode year, excel(myresults, modify)
```

The *force1*, *force2*, and *force3* options

If you try `panelstat` in a dataset that is not a "proper panel" you will obtain an error. This will happen, for example, if there are repeated observations for the unit identifier in the same time period. The options `force1`, `force2`, and `force3` are intended to help you obtain results by ignoring the observations that are causing the problem.

To understand how these options work we will modify the *nlswork* dataset to force an error that will prevent `panelstat` from running. The observations for the first individual in the dataset are

```
. list idcode year if idcode==1

     +---------------+
     | idcode   year |
     |---------------|
  1. |      1     70 |
  2. |      1     71 |
  3. |      1     72 |
  4. |      1     73 |
  5. |      1     75 |
     |---------------|
  6. |      1     77 |
  7. |      1     78 |
  8. |      1     80 |
  9. |      1     83 |
 10. |      1     85 |
     |---------------|
 11. |      1     87 |
 12. |      1     88 |
     +---------------+
```

But, we will force now an error in the dataset by replacing the time value of the first 3 observations by 70.

```
. replace year=70 in 1/3
(2 real changes made)

. list idcode year if idcode==1

     +---------------+
     | idcode   year |
     |---------------|
  1. |      1     70 |
  2. |      1     70 |
  3. |      1     70 |
  4. |      1     73 |
  5. |      1     75 |
     |---------------|
  6. |      1     77 |
  7. |      1     78 |
  8. |      1     80 |
  9. |      1     83 |
 10. |      1     85 |
     |---------------|
 11. |      1     87 |
 12. |      1     88 |
     +---------------+
```

If you now try to run `panelstat` you will obtain an error because this is not a proper panel. You can force `panelstat` to run by ignoring some observations. There are 3 options:

- `force1` - uses the first observation per repeated value so in this case it ignores the second and third observations.
- `force2` - ignore all repeated values, that is, the first three observations.

- `force3` - ignores all observations for the panel unit so, in this case, all observations for which idcode==1.

In our example we only had repeated observations for one individual but if there were multiple individuals with troublesome observations then the same logic would be applied. Note that these options do not modify your dataset – they simply ignore the troubling observations in the ensuing calculations.

The *cont* option

The `cont` option should be used if there is a time gap that is common to all panel units. Looking at the table produced above with the heading "Number of individuals per time unit" we see that some years are missing. If you use the `cont` option then these years will be ignored in all subsequent calculations. This will make a difference in calculations that make use of lagged values.

The *forcestata* and *forcesreshape* option

Working with large panels can be extremely slow. Fortunately, there are some user-written tools such as `gtools` written by Mauricio Caceres, `fastreshape` by Michael Droste and `sreshape` by Kenneth L. Simons. By default `panelstat` will check to see if any of these tools is installed in your machine. If it finds the `gtools` package then it will use it. `fastreshape` is also selected by default but if it cannot find it `panelstat` tries to use `sreshape` instead. Only if the user-written commands are not found will it resort to using *Stata* official commands. However, you can change the behavior of `panelstat`. The option `forcestata` will always use *Stata* official commands while the `forcesreshape` will force the use of `sreshape` even if `fastreshape` is installed. Note, however, that the reshape commands are only used with `panelstat`'s options `tabovert` and `pattern`.

4 The structure of the panel

There are a set of options that allow us to gain a better understanding of the structure of the panel. These are the `pattern`, `gaps`, `runs`,`vars` and `demog` options. In the following, to prevent redisplaying the basic descriptives for the panel we will use the `nosum` option.

The *pattern* option

We start with the `pattern` option

```
. use nlswork, clear
(National Longitudinal Survey.  Young Women 14-26 years of age in 1968)
. panelstat idcode year, pattern nosum

****************************************************
Top 10 patterns in the data
****************************************************

     +--------------------------------+
     |         Pattern    Frequency   |
     |--------------------------------|
  1. | 100000000000000          136   |
  2. | 000000000000001          114   |
  3. | 000000000000111           89   |
  4. | 000000000000011           87   |
  5. | 111111111111111           86   |
     |--------------------------------|
  6. | 000000000011111           61   |
  7. | 110000000000000           56   |
  8. | 000000111111111           54   |
  9. | 000000000001111           54   |
 10. | 000000011111111           49   |
     +--------------------------------+

Note: 1 if observation is in the dataset; 0 otherwise
```

The table that was produced shows the 10 most common patterns in the data. Thus, the most common situation are individuals that were only observed in the first year of the study (136 individuals) while the second most common pattern are individuals observed in the last period (114 individuals). Note that by default we only see the first 10 most common patterns. This behavior can be modified by using the `setmaxpat` option. For example, to report the 25 most common patterns you do

```
panelstat idcode year, pattern setmaxpat(25)
```

The *gaps* option

A gap is simply a "hole" in the data. Option `gaps` gives you some ideia about the gaps in your panel.

```
. panelstat idcode year, gaps nosum
****************************************************
Distribution of the size of the time gaps
****************************************************
    Size of |
  time gaps |      Freq.     Percent        Cum.
------------+-----------------------------------
          1 |      9,757       75.45       75.45
          2 |      1,014        7.84       83.29
          3 |        737        5.70       88.99
          4 |        493        3.81       92.80
          5 |        108        0.84       93.64
          6 |        240        1.86       95.49
          7 |        109        0.84       96.33
          8 |         85        0.66       96.99
```

9	115	0.89	97.88
10	42	0.32	98.21
11	86	0.67	98.87
12	34	0.26	99.13
13	44	0.34	99.47
14	21	0.16	99.64
15	17	0.13	99.77
16	12	0.09	99.86
17	8	0.06	99.92
18	8	0.06	99.98
19	2	0.02	100.00
Total	12,932	100.00	

```
****************************************************
Distribution of the number of gaps by individual
****************************************************
```

Number of gaps per individual	Freq.	Percent	Cum.
0	984	20.89	20.89
1	682	14.48	35.36
2	722	15.33	50.69
3	572	12.14	62.83
4	546	11.59	74.42
5	490	10.40	84.82
6	562	11.93	96.75
7	140	2.97	99.72
8	13	0.28	100.00
Total	4,711	100.00	

The first table that is produced - *Distribution of the size of the time gaps* - gives an idea of how large these gaps are. The most common situation in this data set is a gap of size 1 - which happens 9,757 times. On the other extreme we see that there are 2 gaps of size 19. Since the maximum time dimension of this panel is 21 years this means that two individuals were observed in the first year and again in the last year without having any other observations in between. The second table - *Distribution of the number of gaps by individual* - tells you how many gaps there are per individual. For 984 women there are no gaps - although some of these women may have been observed only once - and, out of the 4,711 women, 13 had a total of 8 gaps!

Notice that there are some years (74, 76, 79, 81, 84 and 86) for which we do not have data. Perhaps there was no data collected in those years, and so this fact may be "inflating" the actual number of gaps. With the `cont` option we can correct this by considering all time periods in the panel as consecutive. If we do this we see that the number of gaps is substantially reduced

```
. panelstat idcode year, gaps nosum cont
****************************************************
Distribution of the size of the time gaps
****************************************************
```

Size of time gaps	Freq.	Percent	Cum.

1	2,093	55.14	55.14
2	673	17.73	72.87
3	340	8.96	81.82
4	227	5.98	87.80
5	167	4.40	92.20
6	116	3.06	95.26
7	67	1.77	97.02
8	49	1.29	98.31
9	28	0.74	99.05
10	14	0.37	99.42
11	12	0.32	99.74
12	8	0.21	99.95
13	2	0.05	100.00
Total	3,796	100.00	

```
***************************************************
Distribution of the number of gaps by individual
***************************************************
```

Number of gaps per individual	Freq.	Percent	Cum.
0	2,171	46.08	46.08
1	1,569	33.31	79.39
2	721	15.30	94.69
3	218	4.63	99.32
4	29	0.62	99.94
5	3	0.06	100.00
Total	4,711	100.00	

If you want you can create variables that capture the number of gaps per panel unit or the size of the largest gap for the panel unit. You do this by adding the option `keepngaps(varname)` or `keepmaxgaps(varname)` where *varname* are names of new variables. For example,

```
. panelstat idcode year, keepngaps(ngap) keepmaxgap(maxgap) nosum cont
```

As a result the variables *ngap* and *maxgap* are added to the dataset. Note that the values for these variables will be different if you do not specify the `cont` option.

The *runs* option

A *run* is a set of consecutive time periods for which a panel unit is observed. If a panel unit has a run of size 3, this means that this panel unit was observed for three consecutive time periods. Of course, a panel unit may have several *runs* (which meant it also had gaps). Next, we run `panelstat` with the `runs` (as well as the `cont`) option

```
. panelstat idcode year, runs nosum cont
***************************************************
Distribution of complete runs by size
```

```
***************************************************
  Length of |
        run |      Freq.     Percent        Cum.
------------+-----------------------------------
          1 |      3,001       35.28       35.28
          2 |      1,635       19.22       54.50
          3 |      1,113       13.08       67.58
          4 |        674        7.92       75.50
          5 |        523        6.15       81.65
          6 |        402        4.73       86.38
          7 |        256        3.01       89.39
          8 |        227        2.67       92.05
          9 |        188        2.21       94.26
         10 |        131        1.54       95.80
         11 |         85        1.00       96.80
         12 |         80        0.94       97.74
         13 |         78        0.92       98.66
         14 |         28        0.33       98.99
         15 |         86        1.01      100.00
------------+-----------------------------------
      Total |      8,507      100.00
```

We have 3,001 runs of size 1 (some of these are *singletons*) while we have 86 women that are observed for 15 consecutive years (ignoring years for which we do not have any data).

The *vars* option

The `vars` option produces a table with information for all numeric variables in your dataset indicating how many panel units fall in each of the following categories

- singleton observation with nonmissing value of the variable
- singleton observation with missing value for the variable
- non-singleton with all missing values of the variable
- non-singleton with only one valid value of the variable
- non-singleton with time-invariant values and nonmissing values for the variable
- non-singleton with time-invariant values and missing values for the variable
- non-singleton with time-variant values and nonmissing values for the variable
- non-singleton with time-variant values and missing values for the variable

Running this option in our dataset we obtain

```
.   panelstat idcode year, vars nosum cont

Distribution of panel units by type of observation for all variables

  ┌──────────────────────────────────────────────────────────────────────────────
──────────────────────────────────┐
  │  variable   s_nonmiss   s_missing   allmissing    onevalue   timeinv_nm   t
> imeinv_wm   timevar_nm   timevar_wm │
  ├──────────────────────────────────────────────────────────────────────────────
```

```
─────────────────────────────────────┤
  │    idcode         547          0            0           8         4164
>         0           0           0 │
  │      year         547          0            0           8            0
>         0        4164           0 │
  │  birth_yr         547          0            0           8         4164
>         0           0           0 │
  │       age         546          1            0          13            3
>         0        4138          18 │
  │      race         547          0            0           8         4164
>         0           0           0 │
  ├──────────────────────────────────────────────────────────────────────────
─────────────────────────────────────┤
  │       msp         547          0            0          10         2112
>         4        2036          10 │
  │   nev_mar         547          0            0          10         2895
>         5        1253           9 │
  │     grade         545          2            0           8         4164
>         0           0           0 │
  │  collgrad         547          0            0           8         4164
>         0           0           0 │
  │  not_smsa         547          0            0           8         3422
>         7         734           1 │
  ├──────────────────────────────────────────────────────────────────────────
─────────────────────────────────────┤
  │    c_city         547          0            0           8         2894
>         5        1262           3 │
  │     south         547          0            0           8         3616
>         6         540           2 │
  │  ind_code         532         15            1          20         1454
>         0        2404         293 │
  │  occ_code         536         11            1          13         1414
>         0        2649          95 │
  │     union         255        292          269         417          527
>      1536         221        1202 │
  ├──────────────────────────────────────────────────────────────────────────
─────────────────────────────────────┤
  │    wks_ue         499         48           18         225          569
>       793        1026        1541 │
  │   ttl_exp         547          0            0           8           22
>         0        4142           0 │
  │    tenure         535         12            0          25           23
>         0        3757         367 │
  │     hours         546          1            0          12          706
>         0        3392          62 │
  │  wks_work         523         24            1          46           26
>         0        3510         589 │
  ├──────────────────────────────────────────────────────────────────────────
─────────────────────────────────────┤
  │   ln_wage         547          0            0           8           13
>         0        4151           0 │
  │    maxgap         547          0            0           8         4164
>         0           0           0 │
  │      ngap         547          0            0           8         4164
>         0           0           0 │
  └──────────────────────────────────────────────────────────────────────────
─────────────────────────────────────┘

s_nonmiss - singleton observation with nonmissing value of the variable
s_missing - singleton observation with missing value for the variable
allmissing - non-singleton with all missing values of the variable
onevalue - non-singleton with only one valid value of the variable
```

```
timeinv_nm - non-singleton with time-invariant values and nonmissing values for
>  the variable
timeinv_wm - non-singleton with time-invariant values and missing values for th
> e variable
timevar_nm - non-singleton with time-variant values and nonmissing values for t
> he variable
timevar_wm - non-singleton with time-variant values and missing values for the
> variable
```

To create a variable with an indicator showing the above cases for a particular variable you need to use the `wiv` option (see below).

The *demog* option

Finally, we discuss the `demog` option. This option characterizes the flows of the panel units that occur between consecutive time periods.

```
. panelstat idcode year, demog nosum
****************************************************
Time changes - incumbents, entrants and exits
****************************************************
```

time	total	inc1	entry	first	reent	inc2	exit	last	reex
68	1375	0	1375	1375	0	851	524	136	388
69	1232	851	381	381	0	1001	231	79	152
70	1686	1001	685	476	209	1315	371	93	278
71	1851	1315	536	381	155	1224	627	156	471
72	1693	1224	469	331	138	1411	282	97	185
73	1981	1411	570	257	313	0	1981	132	1849
75	2141	0	2141	304	1837	0	2141	189	1952
77	2171	0	2171	275	1896	1625	546	163	383
78	1964	1625	339	134	205	0	1964	199	1765
80	1847	0	1847	142	1705	0	1847	143	1704
82	2085	0	2085	159	1926	1647	438	155	283
83	1987	1647	340	126	214	0	1987	239	1748
85	2085	0	2085	147	1938	0	2085	311	1774
87	2164	0	2164	109	2055	1817	347	347	0
88	2272	1817	455	114	341	0	2272	2272	0

```
time - time period
total - total number of individuals at time t
inc1 - number of individuals at t that are also present at t-1
entry - number of individuals at t that are not present at t-1
first - number of individuals at t who show up for the first time at t
reent - number of individuals at t that are reentering at time t
inc2 - number of individuals at t that are also present at t+1
exit - number of individuals at t that are not present at t+1
last - number of individuals at t that are not present at any future time
reexit - number of individuals at t not present at t+1 that appear in later tim
```

```
> es
the following identities hold:
total[t+1]=total[t]-exit[t]+entry[t+1]
inc1=total-entry)
entry=first+reent
inc2=inc1[t+1]
exit=last+reexit
```

If we look at the table we see that in the first year of the panel - 1968 - we had a total of 1,375 individuals. Of those, 851 will be observed again in the following year. Of the remaining 524 which are not observed in 1969, 136 where never again observed in the panel while 388, although not present in 1969, eventually come back in a later year. Moving now to the 1970 row we see that there are 1,686 individuals. 1,001 had been observed in the previous year while 685 are entering the dataset in this year. Of these 685, 476 show up in the data for the first time while 209 had already showed up in a previous year (so they must have gaps). It is no surprise that in 1975 all individuals enter the data - because there was no data in 1974. To account for this we could have specified the `cont` option.

The *all* option

The `all` option is equivalent to simultaneously selecting the options `pattern`, `gaps`, `runs`,`vars` and `demog`

5 Describing your data

`panelstat` offers a set of options that allow you to inspect your variables taking advantage of the panel structure.

The `statovert` option

This option produces descriptive statistics over time for a list of variables. It is quite simple to use. To obtain year by year statistics for the variable *grade* do

```
. panelstat idcode year, nosum statovert(grade)
Descriptive statistics of grade over time
```

time	total	valid	missing	mean	sd	p25	p50	p75
68	1375	1375	0	11.7847	1.89504	12	12	12
69	1232	1232	0	11.8117	1.85027	12	12	12
70	1686	1686	0	11.8909	1.95588	12	12	12
71	1851	1851	0	11.9773	1.91001	12	12	12
72	1693	1693	0	12.0815	1.95619	12	12	12

73	1981	1981	0	12.2024	1.98884	12	12	12
75	2141	2141	0	12.3349	2.20353	12	12	13
77	2171	2171	0	12.5615	2.42971	12	12	14
78	1964	1964	0	12.7032	2.44511	12	12	14
80	1847	1847	0	12.7991	2.44423	12	12	14
82	2085	2085	0	12.8043	2.47387	12	12	14
83	1987	1987	0	12.9461	2.51593	12	12	15
85	2085	2085	0	13.0432	2.47832	12	12	15
87	2164	2164	0	13.1072	2.45129	12	12	15
88	2272	2270	2	13.0969	2.51577	12	12	15

We could have listed several variables in the argument of `statovert`, in which case `panelstat` would create a table for each variable. `statovert` also supports the suboption `detail` which, as suggested by the name, provides additional descriptives on the variables (the 1, 5, 95 and 99th quantile of the variable(s)).

The `wiv` and `wtv` options

The `wiv` option provides statistics for a list of selected variables alongside the panel unit dimension. Using this option on the variables *race* and *union*

```
. panelstat idcode year, nosum wiv(race union)
*****************************************************
*****************************************************
Analyzing variable race within idcode
*****************************************************
There are 100.00% nonmissing observations (28534 out of 28534)
For the variable race we have:
     values range from 1 to 3
     547 singleton idcode-observations with non-missing value (11.61%)
     0 singleton idcode-observations with missing value ( 0.00%)
     0 non-singleton idcode-observations with all values missing ( 0.00%)
     0 non-singleton idcode-observations with only one valid value ( 0.00%)
     4164 non-singleton idcode-observations with year invariant and non-missing
>  values (88.39%)
     0 non-singleton idcode-observations with year invariant and missing values
>  ( 0.00%)
     0 non-singleton idcode-observations with year variant and non-missing valu
> es ( 0.00%)
     0 non-singleton idcode-observations with year variant and missing values (
>  0.00%)
Distribution of all observations for race

                           _wiv_race |      Freq.     Percent        Cum.
-------------------------------------+-----------------------------------
1 singleton non-missing              |        547        1.92        1.92
5 time invariant with non-missing    |     27,987       98.08      100.00
-------------------------------------+-----------------------------------
                               Total |     28,534      100.00
*****************************************************
*****************************************************
Analyzing variable union within idcode
*****************************************************
```

```
There are 67.42% nonmissing observations (19238 out of 28534)
For the variable union we have:
     values range from 0 to 1
     255 singleton idcode-observations with non-missing value ( 5.41%)
     292 singleton idcode-observations with missing value ( 6.20%)
     269 non-singleton idcode-observations with all values missing ( 5.71%)
     409 non-singleton idcode-observations with only one valid value ( 8.68%)
     527 non-singleton idcode-observations with year invariant and non-missing
> values (11.19%)
     1536 non-singleton idcode-observations with year invariant and missing val
> ues (32.60%)
     221 non-singleton idcode-observations with year variant and non-missing va
> lues ( 4.69%)
     1202 non-singleton idcode-observations with year variant and missing value
> s (25.51%)
Distribution of all observations for union
```

_wiv_union	Freq.	Percent	Cum.
1 singleton non-missing	255	0.89	0.89
2 singleton missing	292	1.02	1.92
3 all values missing	745	2.61	4.53
4 one non-missing value	1,310	4.59	9.12
5 time invariant with non-missing	1,986	6.96	16.08
6 time invariant with missing	12,116	42.46	58.54
7 time variant with non-missing	990	3.47	62.01
8 time variant with missing	10,840	37.99	100.00
Total	28,534	100.00	

we obtain a piece of information for each variable. At the top we have information at the level of the individual followed by a table with information at the observation level. Inspecting the output produced by `panelstat` we can see that there are no missing observations for *race*. The panel has 547 singletons (individuals observed only once) but there are no missing values of *race* for the singletons "547 singleton idcode-observations with non-missing value (11.61%)". The remaining 4,164 individual level observations for *race* are all (as expected) time-invariant and no missing values are recorded "4164 non-singleton idcode-observations with year invariant and non-missing values (88.39%)". Singletons represent close to 2% of all observations. The situation is more diverse for the *union* variable. A little less than a third of the observations are missing for the *union* variable. The majority of the singleton observations have missing values for this variable (292 out of 547). There are 269 individuals, observed more than once, that have all observations for *union* missing "269 non-singleton idcode-observations with all values missing (5.71%)" On the other hand, for 409 individuals there is only one value for *union* recorded with the other values missing for the variable. We can see that 527 women have no missing values *and* have maintained the same status over time while 1,536 also show consistent values over time but have some missing values. A smaller number, 221 women, have no missing values in the variable *union* but change status over time. Finally, 1,202 participants see their union status altered over time but have some missing observations. The distribution in terms of observations shows that the largest proportion of the observations (42.5%) are "time-invariant with missing". We could retain in our data a variable indicating the situation of each observation. Running the command with the suboption `keep` as in

```
panelstat idcode year, nosum wiv(race union, keep)
```

would add to the data two variables – *_wiv_race* and *_wiv_union* – to identify the type assigned to each observation.

A less used option is the `wtv` option. It does exactly the same as `wiv` but exchanges the roles of the panel unit and time variable.

The `tabovert` option

This is an option to be used with categorical variables. It simply tabulates the frequency of each category over time. Say, we wanted to understand how the variable union had changed over time. To avoid working with a wide table we restrict the analysis to the period 1968-1977.

```
. panelstat idcode year if year<78, nosum tabovert(union)
```

```
Tabulation of union over time
```

union	n68	n69	n70	n71	n72	n73	n75	n77
0	.	.	620	701	984	817	.	1760
1	.	.	178	227	260	238	.	400
.	1375	1232	888	923	449	926	2141	11

This way we obtain a distribution over time for all values of the variable *union* including missing values.

The `demoby` option

This is also an option to be used only with categorical variables. It can be seen as an extension of the previous option. This option tries to provide some idea about the movements over time of the panel units across the categories of a given variable. Thus, if we run the option on the *union* variable we obtain

```
. panelstat idcode year, nosum demoby(union)
*****************************************************
Decomposition of changes across union over time
*****************************************************
```

time	total	first	last	sing	stay	mover	fmover	rmover
70	798	798	68	68	0	0	0	0
71	928	457	76	49	429	42	42	0
72	1244	477	123	60	660	107	92	15
73	1055	244	121	36	719	92	66	26
77	2160	933	250	125	1006	221	170	51

78	1351	140	133	25	1076	135	96	39
80	1686	233	114	20	1171	282	207	75
82	2083	257	179	34	1513	313	151	162
83	1805	134	221	25	1465	206	83	123
85	2079	212	347	50	1589	278	136	142
87	2161	153	630	60	1699	309	126	183
88	1888	112	1888	112	1562	214	87	127

```
Note: missing values of union are discarded for the analysis (to include specif
> y missing option)
time - time period
total - total number of individuals at time t (total=firs+stay+mover)
first - number of individuals at t that show up for the first time
last - number of individuals at t that show up for the last time
singleton - number of individuals at t that show up only that time (singletons)
stayer - number of individuals at t that were present at the same category of u
> nion since their last observation
mover - number of individuals at t that were present at a different category of
>  union since their last observation (mover=fmover+rmover)
fmover - number of movers at t that are for the first time at that category of
> union
rmover - number of movers at t that are returning to a category of union
```

The above table shows us that in 1970, the first year for which there are nonmissing values of *union*, of the 798 reported values, 68 are for women that only reported values for *union* once. If we move to the row corresponding to the year 1972, we can see that, out of 1,244 valid values reported for the variable *union*, 477 are for women that are reporting their *union* status for the first time, and for 123 women these are the last reported values of *union* status. But there is more information. 660 women maintain the same union status as in the previous year, while 107 have changed their union status. Of those that changed union status, 92 are new to their present category while the other 15 are now returning to a category in which they have been in the past. The suboption `keep` will create a variable identifying each situation at the level of the observation. Thus, if we run

```
panelstat idcode year, nosum demoby(union, keep)
```

the variable *_demoby_union* is created and it will contain an indicator of all possible cases. To understand how this variable is coded consider the list of values for this variable for the first individual in the datase:

```
. list idcode year union _demoby_union if idcode==1
```

	idcode	year	union	_demob~n
1.	1	70	.	.
2.	1	71	.	.
3.	1	72	1	1 first
4.	1	73	.	.
5.	1	75	.	.
6.	1	77	0	3 fmover

7.	1	78	.	.
8.	1	80	1	4 rmover
9.	1	83	1	2 stayer
10.	1	85	1	2 stayer
11.	1	87	1	2 stayer
12.	1	88	1	2 stayer

By default `demoby` does not report information for periods when all values of the variable are missing. That behavior can be overrun with the suboption `missing`

The `flows` option

The option `flows` decomposes the change in the stock of each variable between consecutive periods. Thus, for each time period, it identifies the changes that result from panel units that already exist (incumbent), and those that enter and exit. To illustrate the use of the command we consider the variable *hours* which contains the amount of hours worked by each woman.

```
. panelstat idcode year, nosum flows(hours)
*****************************************************
Time flows for variable hours
*****************************************************
  ┌──────────────────────────────────────────────────────────────────────────
──────┐
  │ time   hours     chg   c_inc   c_exp   c_cont   c_ent   c_exit   c_inc1   c
> _inc2 │
  ├──────────────────────────────────────────────────────────────────────────
──────┤
  │   68   51319       .       0       0        0   51319        .        0
>     0 │
  │   69   46713   -4606     536    2096    -1560   14003   -19112        0
>   -33 │
  │   70   61858   15145     300    1693    -1393   23315    -8512       42
>     0 │
  │   71   67853    5995    1263    3848    -2585   17721   -12989        0
>     0 │
  │   72   61376   -6477     155    3147    -2992   15765   -22376        0
>   -21 │
  ├──────────────────────────────────────────────────────────────────────────
──────┤
  │   73   71490   10114     764    2924    -2160   19065    -9727       12
>     0 │
  │   75   78213    6723       0       0        0   78213   -71490        0
>     0 │
  │   77   78248      35       0       0        0   78248   -78213        0
>     0 │
  │   78   70705   -7543    -133    2824    -2957   11137   -18220       40
>  -367 │
  │   80   66999   -3706       0       0        0   66999   -70705        0
>     0 │
  ├──────────────────────────────────────────────────────────────────────────
──────┤
  │   82   74198    7199       0       0        0   74198   -66999        0
>     0 │
```

```
  |    83   70946   -3252    -281    2686    -2967   11645   -14759      400
>  -257 |
  |    85   76396    5450       0       0        0   76396   -70946        0
>     0 |
  |    87   79711    3315       0       0        0   79711   -76396        0
>     0 |
  |    88   84716    5005     842    3758    -2916   16158   -12232      237
>     0 |
  └──────────────────────────────────────────────────────────────────────────
───────┘

Notes:
hours - total sum of hours at time t
chg - sum of hours at t minus t-1
c_inc - changes from individuals present at t and at t-1 with valid values of h
> ours
  of which:
    c_exp - positive changes (expansions) from individuals present at t and at
> t-1
    c_cont - negative changes (contractions) from individuals present at t and
> at t-1
c_entry - change resulting from entry (present at t but not at t-1)
c_exit - change resulting from exits (present at t-1 but not at t)
c_inc1 - change from individuals present at t and t-1 but with missing data at
> t-1
c_inc2 - change from individuals present at t and t-1 but with missing data at
> t
hours[t]=hours[t-1]+chg, chg=c_inc+c_entry+c_exit+c_inc1+c_inc2, c_inc=c_exp+c_
> cont
```

Consider the second row of the table that was produced. It shows that in 1969 the total amount of hours reported was 46,713 and that represented a decrease of 4,606 when compared with the previous year. Incumbents (those women that were present in 1969 and also in the previous year) increased their working hours by 536. The change in hours worked by incumbents can be decomposed in changes by women that reported an increase in hours worked (2,096 hours) and women that decreased the amount of hours worked (1,560 hours). The next column shows the amount of hours added by women that entered the survey in 1969 (were not present in the previous year) - a total contribution of 14,003 while women that exited from 1968 to 1969 contributed with a change of -19,112 hours. We also see that there are no incumbents with missing data in 1968 but there is a reduction of 33 hours which is accounted for by missing data of incumbent(s) who had hours reported in 1968. You can specify several variables and a table will be produced for each variable.

If you want to know how many panel units are behind the numbers produced by the option `flows` you need to specify the *unit* option. That is, you should instead do

```
panelstat idcode year, nosum flows(hours, unit)
```

5 Inspecting your data

This set of commands is particularly helpful to spot problems in the data.

The `rel` and `abs` options

These options report on changes over time for the specified variable. The options are particularly suited for continuous variables. As an example we will use the `rel` option with the variable *hours*

```
. panelstat idcode year, nosum rel(hours) cont

*****************************************************
Relative changes over time for hours (threshold set to 100)
*****************************************************

     _rel_L1_hours |      Freq.     Percent        Cum.
-------------------+-----------------------------------
1 positive change  |      4,450       15.60       15.60
2 negative change  |      3,917       13.73       29.32
3 no change        |     11,092       38.87       68.20
4 abnormal pos chg |        278        0.97       69.17
5 abnormal neg chg |        200        0.70       69.87
6 missing          |      8,597       30.13      100.00
-------------------+-----------------------------------
             Total |     28,534      100.00
Note: Relative change is calculated with respect to the average of x_{t} and x_
> {t-1}
```

The output classifies all relative changes from two consecutive periods for the same panel unit and tabulates the results. It distinguishes between no change, positive and negative change, and abnormal positive, and abnormal negative changes. Abnormal changes are those that exceed a specified threshold value for a relative change (by default 100). You can change the threshold value using the suboption `val`. For example, if we try

```
. panelstat idcode year, nosum rel(hours, val(50)) cont

*****************************************************
Relative changes over time for hours (threshold set to 50)
*****************************************************

     _rel_L1_hours |      Freq.     Percent        Cum.
-------------------+-----------------------------------
1 positive change  |      3,792       13.29       13.29
2 negative change  |      3,411       11.95       25.24
3 no change        |     11,092       38.87       64.12
4 abnormal pos chg |        936        3.28       67.40
5 abnormal neg chg |        706        2.47       69.87
6 missing          |      8,597       30.13      100.00
-------------------+-----------------------------------
             Total |     28,534      100.00
Note: Relative change is calculated with respect to the average of x_{t} and x_
> {t-1}
```

we see that the number of abnormal positive (and negative) changes increases because we are now classifying as abnormal a relative change exceeding 50%. We should also note that, by default, relative changes are calculated with respect to the average of starting and end point. This behavior can be changed with the `denlag` suboption. If that option is specified then the relative change is calculated with respect to the lagged value. We can also change the number of lags used on the calculation of the relative change. By default that value is 1. But you could specify a different value with the option `lags`. Finally, suboption `keep` creates a variable that stores the classification of type of change attributed to the observations of the variable. The command

```
panelstat idcode year, nosum rel(hours, keep) cont
```

would create a variable named $\_rel\_L1\_hours$.

The option `abs` operates similarly but it classifies the absolute changes. The threshold change to report abnormal changes is 10 and may be changed with the suboption `val`. As with the option `rel` you can use a different value for the lag (the default is 1). This is done with suboption `lags`. If using a lag larger than one you may prefer to use differences in differences. In that case you need to specify suboption `dif`. Finally, you can use the `keep` option to retain the variable with the classifications.

The `fromto` option

This is another option intended for categorical variables. Basically, for a given variable, it tabulates all combinations of values at two different time periods. The `cont` option is ignored if you specify the `fromto` option because you have to explicitly identify the time periods. Thus, if I wanted to find out the changes in variable *union* from 87 to 88 I would do

```
. panelstat idcode year, nosum fromto(union, from(87) to(88) )

Change of union from 87 to 88
```

union87	union88	_type	n
0	1	4 dif	80
1	0	4 dif	75
0	0	3 same	1072
1	1	3 same	304

The results show that from 1987 to 1988, 80 women became unionized while 75 left the union. 1072 remained unionized while 304 remained unionized. If we add the suboption `missing` then the missing observations of *union* are also accounted for:

```
. panelstat idcode year, nosum fromto(union, from(87) to(88) missing )

Change of union from 87 to 88
```

union87	union88	_type	n
0	1	4 dif	80
1	0	4 dif	75
0	0	3 same	1072
1	1	3 same	304
.	0	2 entry	276
.	1	2 entry	81
0	.	1 exit	527
1	.	1 exit	103

The `from` argument is required and it must contain a valid value for the time variable, while the `to` argument may be omitted. If that is the case then it is assumed that the change is for the following time period. Thus, the same results would obtain had we specified

```
panelstat idcode year, nosum fromto(union, from(87) missing )
```

If you want the displayed table to be sorted from lowest to highest frequency you can use the `ascend` option while the `descend` option does the reverse. `fromto` also supports two other suboptions – `save` and `keep`. The `save` will save a *Stata* file with the results shown in the table, while the `keep` option adds a variable to the dataset that identifies for each observation the following situations:

- 0 not flagged - observation was not considered
- 1 exit - missing value of the variable at *to*
- 2 entry - missing value of the variable at *from*
- 3 same - values are the same
- 4 dif - values are different

In this case the variable name would be $__ft_union_$87_88.

The `return` option

This is another option intended to look at the change within panel unit of the values of a specific variable. It considers three time periods say x_t, x_{t+1}, and x_{t+2}. It checks whether the value of x_t and x_{t+2} are identical but x_{t+1} differs. If we try

```
. panelstat idcode year, nosum return(union, from(71) middle(72) to(73) )
```

```
Return of union from 71 to 73
```

union71	union72	union73	n
0	1	0	12
1	0	1	4

then we are requesting `panelstat` to identify observations of union for which values of 71 and 73 are identical with differing values in 72. We see that there are 12 women with a “0 1 0” sequence and another 4 with a “1 0 1” sequence. These may signal coding errors. Since we have three consecutive time values we can simply code

```
panelstat idcode year, nosum return(union, from(71))
```

to obtain the same result. But remember that the arguments for the `from`, `middle` and `to` suboptions must be valid time values. The suboption `save` will save the table of results to a Stata file. You can also use the `keep` suboption to add a variable to the dataset that identifies the flagged observations. In this case the variable would be named *_ret_union*_71_73. The `return` option can also be used with continuous variables. In that case you need to specify the `within` suboption. With this suboption the command will look for cases where x_{t+1} is outside the interval $[x_t * (1 - a\%), x_t * (1 + a\%)]$ where a is the value used in the argument of the `within` suboption. To illustrate let us check for abnormal changes in wages for 72.

```
. panelstat idcode year, nosum return(ln_wage, from(71) within(50))

Return of ln_wage from 71 to 73
```

ln_wa~71	ln_wa~72	ln_wa~73	n
.8007324	1.292054	.9011876	1
.8462984	1.309547	1.219973	1
.8462984	1.503363	1.231429	1
.9416081	1.454573	.9845695	1
.9830538	1.483829	1.321042	1
1.023876	1.655147	1.480504	1
1.115543	.5257036	1.166891	1
1.206198	.2506002	1.079615	1
1.217862	1.860038	1.394327	1
1.316302	.28339	1.072743	1
1.328725	.3226964	1.525354	1
1.376927	.5522707	.9437474	1
1.434085	2.249825	2.148098	1
1.472237	.3903617	.8322071	1
1.49388	2.370864	1.503363	1
1.790204	.3609478	1.283395	1
1.835839	3.02527	2.707336	1

The above table gives a list of all cases where *ln_wage* in 72 is outside a 50% percent interval constructed around the 71 value while the 73 value is within that interval.

The `trans` option

Again this is an option meant to help you identify potential problems in the data. It is meant to be used with categorical variables. The idea is simple. For all units in each category of the variable in year *t* it calculates the share of those same units that came from each different category of the variable at *t-1*. We call this the *transition probabilities*. Thus, if in a given year a panel unit has a transition probability of 100% it means that all individuals that belong to the same category at year *t* also belonged to the same category at *t-1* (but the categories in *t* and *t-1* need not be the same). Likewise, a transition probability of 10% means that 10% of individuals in a given category at year *t* came from the same category where they were classified in *t-1*. The results are presented in a table that shows, for each time period, the number of panel units grouped into 4 categories of the *transition probabilities*. To exemplify, let us apply this option to the variable age

```
. panelstat idcode year, nosum trans(age, keep)

*****************************************************
Distribution of transition probabilities (t-1 to t) for classes of age
*****************************************************
```

Time	Distribution of probabilities p<25	25<=p<75	75<=p<100	p=100	Total
69	48	0	765	31	844
70	48	0	936	17	1,001
71	29	0	1,219	67	1,315
72	41	0	1,071	112	1,224
73	58	0	1,319	34	1,411
78	90	2	1,426	107	1,625
83	71	0	1,536	40	1,647
88	1	1,763	41	0	1,805
Total	386	1,765	8,313	408	10,872

The option `keep` adds the variable *_trans_age* where the transition probabilities are stored. The results are summarized in the table. If we look at the last year, 1988, we can see that we were able to compute transition probabilities for 1,805 individuals (these were individuals with valid values for age in 87 and 88). One individual has a transition probability lower than 25%. Digging into the data we find that the idcode for this individual is

```
. list idcode if _trans_age<25 & year==88

        +--------+
        | idcode |
        |--------|
19224.  |   3462 |
        +--------+
```

and if we list the observations for this individual we obtain

```
. list idcode year age if idcode==3462
```

```
     |  idcode   year   age
     |--------------------------
19218. |    3462     75    20
19219. |    3462     78    23
19220. |    3462     82    27
19221. |    3462     83    28
19222. |    3462     85    30
     |--------------------------
19223. |    3462     87    32
19224. |    3462     88    34
```

This subject was flagged because it was aged 32 in 1987 and 34 in 1988. It was the only individual out of 42 with valid age values in 1987 and 1988 that moved from age 32 in 1987 to age 34 in 1988 (a “transition probability” of 1/42=2.38%).

The suboption `missing` also accounts for transitions from missing to a valid value of the variable. In that case we would have to specify

```
panelstat idcode year, nosum trans(age, miss)
```

We can use the suboptions `low` and `upper` to define the threshold levels used in the table that is displayed. If, say, we want to find out how many women, each year, had transition probabilities below 1% and above 99% we could write:

```
. panelstat idcode year, nosum trans(age, low(1) upper(99))

****************************************************
Distribution of transition probabilities (t-1 to t) for classes of age
****************************************************
```

Time	Distribution of probabilities p<1	1<=p<99	99<=p<100	p=100	Total
69	0	813	0	31	844
70	1	983	0	17	1,001
71	3	1,138	107	67	1,315
72	4	1,108	0	112	1,224
73	1	1,376	0	34	1,411
78	3	1,515	0	107	1,625
83	3	1,604	0	40	1,647
88	0	1,805	0	0	1,805
Total	15	10,342	107	408	10,872

The quantr option

The `quantr` option is also intended to help find problems in the data. However, it is intended for use with continuous data. As an example, we use this option with the *ln_wage* variable. For each year, we compute the 25th, 50th and 75th percentile. Next we look, for consecutive years, how individuals move between these percentiles. Thus, if we do,

```
. panelstat idcode year, nosum quantr(ln_wage)

****************************************************
```

```
changes (t-1 to t) in the quartiles of ln_wage
****************************************************
           |                                         Distribution of quantile changes
      Time |       1to1        1to2        1to3        2to1        2to2        2to3
>     3to1         3to2        3to3 |      Total
-----------+------------------------------------------------------------------------
----------------------------------+---------
        69 |        106          63          16          73         304          56
>        7           49         177 |        851
        70 |        174         135          13          51         435         102
>        8           55         237 |      1,210
        71 |        185         145          21         103         524         105
>       13           76         298 |      1,470
        72 |        172         129          13         108         504          90
>        8           58         280 |      1,362
        73 |        269         150          15         114         657          92
>       14           68         345 |      1,724
        75 |        261         213          29         167         597         148
>       31          102         289 |      1,837
        77 |        279         155          19         178         680         101
>       24          129         331 |      1,896
        78 |        307         120          17         149         679          77
>       12          119         350 |      1,830
        80 |        264         146          11         156         608         107
>       18           99         296 |      1,705
        82 |        339         175          15         130         677         134
>       21          108         327 |      1,926
        83 |        328          89          10         133         733          82
>       19           96         371 |      1,861
        85 |        318         141          15         154         715          93
>       13          135         354 |      1,938
        87 |        347         159           7         146         782         129
>       18           87         380 |      2,055
        88 |        383         147          16         115         837         130
>       29          104         397 |      2,158
-----------+------------------------------------------------------------------------
----------------------------------+---------
     Total |      3,732       1,967         217       1,777       8,732       1,446
>      235        1,285       4,432 |     23,823
Notes:
 quartile 1 defined as values below 25
 quartile 2 defined as values above 25 and below 75
 quartile 3 defined as values above 75
```

we can find the year-to-year movements across the percentiles of *ln_wage*. Of the 851 individuals in 1969 (those that had non missing wage values in that and the previous year) 106 moved from the first quartile in 1968 to the first quartile in 1969 (1to1 column). More relevant are probably the 16 individuals that had a wage above the 3rd quartile in 1969 but a wage below the 1st quartile in 1968 (1to3 column) If we want we can show the table in terms of shares. For this we use the suboption `rel` as in,

```
. panelstat idcode year, nosum quantr(ln_wage, rel)

****************************************************
changes (t-1 to t) in the quartiles of ln_wage
****************************************************
           |                                         Distribution of quantile changes
```

```
      Time |       1to1       1to2       1to3       2to1       2to2       2to3
>     3to1       3to2       3to3 |     Total
-----------+------------------------------------------------------------------
-----------+-----------------------------------+
        69 |      12.46       7.40       1.88       8.58      35.72       6.58
>     0.82       5.76      20.80 |    100.00
        70 |      14.38      11.16       1.07       4.21      35.95       8.43
>     0.66       4.55      19.59 |    100.00
        71 |      12.59       9.86       1.43       7.01      35.65       7.14
>     0.88       5.17      20.27 |    100.00
        72 |      12.63       9.47       0.95       7.93      37.00       6.61
>     0.59       4.26      20.56 |    100.00
        73 |      15.60       8.70       0.87       6.61      38.11       5.34
>     0.81       3.94      20.01 |    100.00
        75 |      14.21      11.59       1.58       9.09      32.50       8.06
>     1.69       5.55      15.73 |    100.00
        77 |      14.72       8.18       1.00       9.39      35.86       5.33
>     1.27       6.80      17.46 |    100.00
        78 |      16.78       6.56       0.93       8.14      37.10       4.21
>     0.66       6.50      19.13 |    100.00
        80 |      15.48       8.56       0.65       9.15      35.66       6.28
>     1.06       5.81      17.36 |    100.00
        82 |      17.60       9.09       0.78       6.75      35.15       6.96
>     1.09       5.61      16.98 |    100.00
        83 |      17.62       4.78       0.54       7.15      39.39       4.41
>     1.02       5.16      19.94 |    100.00
        85 |      16.41       7.28       0.77       7.95      36.89       4.80
>     0.67       6.97      18.27 |    100.00
        87 |      16.89       7.74       0.34       7.10      38.05       6.28
>     0.88       4.23      18.49 |    100.00
        88 |      17.75       6.81       0.74       5.33      38.79       6.02
>     1.34       4.82      18.40 |    100.00
-----------+------------------------------------------------------------------
-----------+-----------------------------------+
     Total |      15.67       8.26       0.91       7.46      36.65       6.07
>     0.99       5.39      18.60 |    100.00
Notes:
 quartile 1 defined as values below 25
 quartile 2 defined as values above 25 and below 75
 quartile 3 defined as values above 75
```

We can, if we want, redefine the cut-off percentiles used to define the quartiles. For this we use the suboptions `low` and `upper`, as in

```
. panelstat idcode year, nosum quantr(ln_wage, low(10) upper(90))

******************************************************
changes (t-1 to t) in the quartiles of ln_wage
******************************************************
           |                                 Distribution of quantile changes
      Time |       1to1       1to2       1to3       2to1       2to2       2to3
>     3to1       3to2       3to3 |     Total
-----------+------------------------------------------------------------------
-----------+-----------------------------------+
        69 |         23         38          0         32        632         35
>        2         24         65 |       851
        70 |         48         74          0         37        886         48
>        1         30         86 |     1,210
        71 |         55         86          3         62      1,053         56
>        3         46        106 |     1,470
```

```
        72 |        48         84          3         58        983         58
>        1         42         85 |      1,362
        73 |        77         98          3         79      1,259         43
>        3         32        130 |      1,724
        75 |        49        145          3        121      1,265         82
>        6         67         99 |      1,837
        77 |        69        101          1        117      1,343         62
>        7         81        115 |      1,896
        78 |        67         89          6        121      1,317         57
>        1         62        110 |      1,830
        80 |        73         87          3        172      1,144         67
>        4         62         93 |      1,705
        82 |       106        178          4        100      1,273         81
>        2         70        112 |      1,926
        83 |        78         99          2         99      1,340         48
>        4         63        128 |      1,861
        85 |        76        103          4        113      1,388         66
>        1         69        118 |      1,938
        87 |        94         98          1        108      1,483         75
>        2         62        132 |      2,055
        88 |       120        107          8         98      1,534         82
>        3         79        127 |      2,158
-----------+--------------------------------------------------------------------
-------------------------------------------+-----------
     Total |       983      1,387         41      1,317     16,900        860
>       40        789      1,506 |     23,823
Notes:
 quartile 1 defined as values below 10
 quartile 2 defined as values above 10 and below 90
 quartile 3 defined as values above 90
```

Now, quartile 1 would correspond to all individuals with wages up to the 10th percentile, while the quartile 3 would correspond to all individuals with wages above the 90th quartile.

The tables created by the `quantr` option ignore the transitions that originate from missing values. For example, the values reported in 1969 ignore individuals that had missing wage data in 1968. We can report these values by adding the `missing` option as in

```
panelstat idcode year, nosum trans(ln_wage, missing)
```

Finally, as with several other options, we may add a variable to the data set that contains information about the case that applies to each observation. This is done using the suboption `keep`.

Miscellaneous

The `checkid` option

This option is used if you have an alternative identifier for the panel unit var and want to find out if how close that variable is to your known panel unit identifier. For example, suppose that in this dataset, besides the *idcode* you had available another variable – say the tax id number (*taxid*). To understand how close *taxid* is to a panel unit identifier you can compare it to *idcode*. This is done by running the `checkid` option as in

```
panelstat idcode year, checkid(taxid)
```

This option will produce a table with the number of panel units that fit in each of the following cases:

- 1 - 1:1 ids coincide - idcode and taxid coincide
- 2 - 1:m multiple values of taxid - one idcode corresponds to multiple values of taxid
- 3 m:1 multiple values of id - there are multiple idcodes with the same taxid
- 4 m:m multiple values of taxid and id - there are multiple idcodes mixed with multiple taxid
- 5 1:. all values missing for taxid " - one idcode with all values missing for taxcode
- 6 1:.1 unique values of taxid with missing - one idcode with unique values of taxid but with missing values
- 7 1:.m multiple values of taxid with missing - one idcode with multiple values of taxid and missing
- 8 m:. multiple values of id with missing - multiple values of idcode with multiple taxids and missing

You can create a variable that stores each one of these cases using the suboption `keep`. Note that to run this option you need to previously install `group2hdfe` a Stata user-written command available at SSC. If not already installed simply type

```
ssc install group2hdfe
```

at the Stata prompt.

6 Acknowledgements

I am grateful for all the comments, suggestions, and bug corrections by the BPlim staff, particularly Marta Silva and Emma Zhao. Other researchers, such as Ana Rute Cardoso, have also provided valuable suggestions.