

Paper SD-106

Identifying Gaps in Time Series Data

Bruce Gilsen, Federal Reserve Board, Washington, DC

ABSTRACT

Missing observations are a common issue with time series data. For a small amount of data, you can print the data set and inspect manually, but this is not realistic for large data sets, especially if there are multiple BY groups.

With the TIMEID procedure, which is part of SAS/ETS® software, you can easily check a data set to determine if observations are missing, and print a simple report that shows the location and quantity of missing observations.

PROC TIMEID can be used with SAS® date or datetime variables, and for any frequency. The examples in this paper use SAS dates with frequency WEEKDAY, which is common at the Federal Reserve Board.

Example 1. Data without a BY variable

Data set ONE contains WEEKDAY data, and DATE is a SAS date. Note that

- DATE, INCOME, INVEST, GOV, and TAXES are numeric variables. DATE contains SAS dates.
- April 17, 1995 is a Monday.
- 2 dates are missing after Tuesday, April 25.
- 1 date is missing after Friday, April 28.
- 4 dates are missing after Wednesday, May 3.

Obs	date	income	invest	gov	taxes
1	19950417	1	2	3	4
2	19950418	5	6	7	8
3	19950419	9	10	11	12
4	19950420	13	14	15	16
5	19950421	17	18	19	20
6	19950424	21	22	23	24
7	19950425	25	26	27	28
8	19950428	29	30	31	32
9	19950502	33	34	35	36
10	19950503	37	38	39	40
11	19950510	41	42	43	44

Sort the data by the time ID variable if it is not already sorted that way. In this case, the sort shown here can be omitted.

```
proc sort data=one;
  by date;
run;
```

Now, PROC TIMEID examines data set ONE and creates data set OUTINT with information about missing observations. SAS dates are displayed with the YYMMDDN8. format.

```
proc timeid data=one outinterval=details=outint;
  id date interval=weekday format=yyymmddn8.;
run;
```

In SAS 9.3 and 9.4, the output data set has one observation and 38 variables. For readability, let's just print some useful variables.

```
proc print data=outint;
  var timeid start end nspans minspan maxspan msg interval;
run;
```

Variable	Value	Description
TIMEID	date	Name of the time ID variable
START	19950417	The smallest (earliest) time ID
END	19950510	The largest (latest) time ID
NSPANS	4	The number of spans in the data (1 = all values are contiguous, 2 = one gap in the data, 3 = 2 gaps in the data, etc.)
MINSPAN	1	Minimum number of periods between time ID values
MAXSPAN	5	Maximum number of periods between time ID values
MSG	(see below)	A status message that is also printed to the SAS log
INTERVAL	WEEKDAY	The time interval that was specified

MSG value: WARNING: Interval consistent but data contains gaps.

Now, use the ODS OUTPUT statement with PROC TIMEID to create data set DECOMP with information about missing observations. The ODS OUTPUT statement specifies which table (component of the results) is written to a SAS data set. The Printed Tabular Output section of the PROC TIMEID documentation has a list of ODS table names for PROC TIMEID.

The ODS OUTPUT statement is only in effect for the procedure that follows it. ODS OUTPUT statement syntax is

```
ods output table-name = data-set-name;
```

In this case, the ODS OUTPUT statement creates a data set that includes the following variables.

- TIME is a SAS date or datetime containing the values of the time ID variable. We'll rename it to DATE in the next step for readability since that was the name of our time ID variable.
- SPAN is the number of intervals between the current and previous values of TIME.
 - o If one, observations are contiguous.
 - o If two, then one period is missing between the previous and current observations.
 - o If three, then two periods are missing between the previous and current observations.
 - o etc.
- COUNT is the number of occurrences of the current value of TIME. If there are no repeat dates, it is 1 in all cases.

```
ods output Decomposition=decomp;
proc timeid data=one outintervaldetails=outint print=values;
  id date interval=weekday format=yymmddn8.;
run;
```

I'm more interested in the number of missing observations than the interval size. For example, the interval between observations 7 (19950425) and 8 (19950428) is 3 periods, but I want to know that there are 2 missing periods between these observations. So, I subtract 1 from SPAN. In observations that are contiguous to the previous observation, SPAN is now 0. SPAN is missing in observation 1 so the IF statement avoids a warning message.

```
data decomp;
  set decomp (rename=(time=Date));
  if _n_ ne 1 then span=span-1;
run;
```

Now, print observations that are not contiguous to the previous observation. By applying the WHERE clause to the data set rather than subsetting in a previous step, the output includes the observation numbers of the complete data set, which is useful for finding locations where data are missing.

```
proc print data=decomp;
  var date span count;
  where span>0;
run;
```

Output is as follows.

Obs	Date	Span	Count
8	19950428	2	1
9	19950502	1	1
11	19950510	4	1

We can easily see that observations are missing as follows.

- 2 dates are missing before April 28, 1995, which is observation 8.
- 1 date is missing before May 2, 1995, which is observation 9.
- 4 dates are missing before May 10, 1995, which is observation 11.

Example 2. Data with a BY variable

In many cases, time series is organized in BY groups. Let's create and then sort data set TWO, which includes two BY groups.

As before, DATE, INCOME, INVEST, GOV, and TAXES are numeric variables. DATE contains SAS dates and April 17, 1995 is a Monday.

- For BANKID = 1:
 - o We have the same data values as in Example 1.
 - o 2 dates are missing after Tuesday, April 25.
 - o 1 date is missing after Friday, April 28.
 - o 4 dates are missing after Wednesday, May 3.
- For BANKID = 2:
 - o We have the same data values as in Example 1 except for an additional observation, for May 1.
 - o 2 dates are missing after Tuesday, April 25.
 - o 4 dates are missing after Wednesday, May 3.

Obs	bankid	date	income	invest	gov	taxes
1	1	19950417	1	2	3	4
2	1	19950418	5	6	7	8
3	1	19950419	9	10	11	12
4	1	19950420	13	14	15	16
5	1	19950421	17	18	19	20
6	1	19950424	21	22	23	24
7	1	19950425	25	26	27	28
8	1	19950428	29	30	31	32
9	1	19950502	33	34	35	36
10	1	19950503	37	38	39	40
11	1	19950510	41	42	43	44

12	2	19950417	1	2	3	4
13	2	19950418	5	6	7	8
14	2	19950419	9	10	11	12
15	2	19950420	13	14	15	16
16	2	19950421	17	18	19	20
17	2	19950424	21	22	23	24
18	2	19950425	25	26	27	28
19	2	19950428	29	30	31	32
20	2	19950502	33	34	35	36
21	2	19950503	37	38	39	40
22	2	19950510	41	42	43	44
23	2	19950501	45	46	47	48

```
proc sort data=two;
  by bankid date;
run;
```

PROC TIMEID examines data set TWO and creates data set OUTINT with information about missing observations.

```
proc timeid data=two outintervaldetails=outint;
  id date interval=weekday format=yymmddn8.;
  by bankid;
run;
proc print data=outint;
  var bankid timeid start end nspans minspan maxspan msg interval;
run;
```

OUTINT contains 1 observation for each BY group.

Obs	bankid	TIMEID	START	END	NSPANS	MINSPAN	MAXSPAN
1	1	date	19950417	19950510	4	1	5
2	2	date	19950417	19950510	3	1	5

Obs	MSG	INTERVAL
1	WARNING: Interval consistent but data contains gaps.	WEEKDAY
2	WARNING: Interval consistent but data contains gaps.	WEEKDAY

As in Example 1, PROC TIMEID with the ODS OUTPUT statement creates data set DECOMP with information about missing observations, and after subtracting 1 from SPAN it contains the number of missing periods between the current and previous observations. The IF statement avoids warning messages in the first observation of each BY group, when SPAN is missing.

```
ods output Decomposition=decomp;
proc timeid data=two outintervaldetails=outint print=values;
  id date interval=weekday format=yymmddn8.;
  by bankid;
run;

data decomp;
  set decomp (rename=(time=Date));
  by bankid;
  if not first.bankid then span=span-1;
run;
proc print data=decomp;
```

```
var bankid date span count;
where span>0;
run;
```

Output is as follows.

Obs	bankid	Date	Span	Count
8	1	19950428	2	1
9	1	19950502	1	1
11	1	19950510	4	1
19	2	19950428	2	1
23	2	19950510	4	1

We can easily see that observations are missing as follows.

- 2 dates are missing before April 28, 1995, which is observation 8.
- 1 date is missing before May 2, 1995, which is observation 9.
- 4 dates are missing before May 10, 1995, which is observation 11.
- 2 dates are missing before April 28, 1995 for BANKID 2, which is observation 19.
- 4 dates are missing before May 10, 1995 for BANKID 2, which is observation 23.

A macro to generalize the code

Macro TSGAPS generalizes the code in the previous examples. It contains the code used in the previous two examples, with a few small additions.

- Conditional macro code generates slightly different SAS code for cases with and without BY variables.
- The final data set is checked to see if any observations are missing. If so, the report detailing missing observations is printed. Otherwise, a note is written to the SAS log and no report is printed.

Macro TSGAPS has four required arguments and two optional arguments, as follows.

```
%TSGAPS(data=, out=, timeid=, interval=<, byvar=><, format=>);
```

Required arguments

data=*input-dataset*

The data set to examine

out=*output-dataset*

The output data set containing the final results

timeid=*timeid-variable*

The time ID variable: a SAS date or datetime

interval=*interval-value*

The periodicity of the time ID variable

Optional arguments

byar=*byvar-values*

A list of BY variables if any

format=format-value

Format used to display the time ID variable, defaults to YYMMDDN8.

Here is the macro.

```
%macro tsgaps(data=, out=, timeid=, interval=, byvar=, format=yyymmddn8.);

  %local max_span;

  /* Verify that required arguments were provided */
  %if &data = or &out = or &timeid = or &interval = %then %do;
    %put ERROR: Macro TSGAPS aborts because required argument not provided.;
  %end;
  %else %do;

    /* Sort by the BY variables if any, and by the Time ID variable */
    proc sort data=&data;
      by &byvar &timeid;
    run;

    /* Examine the data set for missing observations and print some results */
    proc timeid data=&data outintervaldetails=outint;
      id &timeid interval=&interval format=&format;
      %if &byvar ne %then %do;
        by &byvar;
      %end;
    run;

    proc print data=outint;
      var &byvar timeid start end nspans minspan maxspan msg interval;
    run;

    /* Create a SAS data set with information about missing observations */
    ods output Decomposition=&out;
    proc timeid data=&data outintervaldetails=all print=values;
      id &timeid interval=&interval format=&format;
      %if &byvar ne %then %do;
        by &byvar;
      %end;
    run;

    /* Modify the data so that SPAN has the number of missing observations
       between the current and previous observation. Print a summary
       report that shows the missing observations.
    */
    data &out;
      set &out (rename=(time=&timeid));
      %if &byvar ne %then %do;
        by &byvar;
        if not first.&byvar then span=span-1;
      %end;
      %else %do;
        if _n_ ne 1 then span=span-1;
      %end;
    run;
```

```

/* Determine largest value of SPAN and copy it to macro variable MAX_SPAN */
proc means data=&out noprint;
  var span;
  output out=_&out max=maxspan;
run;
proc sql noprint;
  select maxspan into :max_span trimmed
  from _&out;
quit;

/* If maximum value of SPAN is 0, then no observations are missing.
   Otherwise, print a missing observation summary report. */
%if &max_span = 0 %then %do;
  %put NOTE: No observations in data set &data are missing;
%end;
%else %do;
  proc print data=&out;
    var &byvar &timeid span count;
    where span>0;
  run;
%end;
%end;
%mend tsgaps;

```

Here is how to invoke the macro for the examples in this paper.

For example 1:

```
%tsgaps(data=one, out=decomp1, timeid=date, interval=weekday);
```

For example 2:

```
%tsgaps(data=two, out=decomp2, timeid=date, interval=weekday, byvar=bankid);
```

Beyond the basics

The code in this paper shows a small subset of the capabilities of PROC TIMEID. It also reflects the personal preferences of the author, for example, to see the number of missing periods rather than the number of periods between two observations. I encourage you to read the PROC TIMEID documentation and design code that provides the information most useful to you.

CONCLUSION

PROC TIMEID, which is part of SAS/ETS software, can be used to obtain information about missing observations in time series data, which is a common problem. The paper contained a few simple examples, and the PROC TIMEID documentation includes information about other features that might be useful.

REFERENCES

Beeman, Jennifer (2014). "*Tips: Identifying and Locating Missing Values and Gaps in Time Series Data*," available at <https://communities.sas.com/docs/DOC-9932>.

SAS Institute Inc. (2011), "*SAS/ETS(R) 9.3 User's Guide*," Cary, NC: SAS Institute Inc.

ACKNOWLEDGMENTS

This paper was heavily influenced by Jennifer Beeman's SAS Communities article on PROC TIMEID (Beeman, 2014). The code used in Example 1 is taken from code in that article.

In addition, the following people contributed extensively to the development of this paper: Heidi Markovitz and Donna Hill at the Federal Reserve Board.

CONTACT INFORMATION

For more information, contact the author, Bruce Gilsen, by mail at Federal Reserve Board, Mail Stop N-122, Washington, DC 20551; by e-mail at bruce.gilsen@frb.gov; or by phone at 202-452-2494.

TRADEMARK INFORMATION

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.