Time Series Magic: Using PROC EXPAND with Time Series Data

Andrew H. Karp
Sierra Information Services, Inc.
Sonoma, California USA

Abstract

PROC EXPAND is a little-known but very powerful SAS/ETS™ Software Procedure that contains many tools for working with time series data. Taking the time to master its capabilities is a worthwhile return on your investment as a SAS® Software user, as its features often obviate the need for complex data step coding to work with your time series data. In addition, PROC EXPAND contains a number of tools for working with time series data that are not available in other parts of SAS System software.

PROC EXPAND is part of the Econometrics and Time Series (ETS) module of the SAS System. It was first introduced in SAS Version 6.0 and has been enhanced considerably in subsequent releases of the SAS System. PROC EXPAND creates output data sets, it does not create “printed output,” nor does it send results to the Output Delivery System (ODS) for subsequent rendering as Output Objects. At least one variable (column) in data sets used by PROC EXPAND must be a SAS date, time, or datetime variable, and that variable must “form a time series.”

PROC EXPAND Functionalities

PROC EXPAND can carry out the following tasks on your time series data:

- **Aggregate** a time series from a higher to a lower sampling frequency
- **Interpolate** higher sampling frequency observations from data observed/colllected at a lower sampling frequency
- **Replace missing values** in a time series with values estimated by fitting a cubic spline function to the non-missing data (default) or by other optional interpolation methods documented in the PROC EXPAND Chapter of SAS/ETS documentation
- **Transform** values of the time series by applying transformation operators to the time series, either prior to or after aggregation/interpolation of the input series. Among the wide range of transformation operators available in PROC EXPAND are moving time window statistics such as moving sums, moving averages, etc.
- **Change** the observational characteristics of the input series. [This capability is not addressed in this paper, please see the PROC EXPAND documentation for details].

The results of PROC EXPAND’s “work” on your data are stored in either a permanent or temporary SAS data set. No output is sent to the Output Window.

PROC EXPAND can therefore be considered a “housekeeping” procedure, as it is often used to “clean up” or prepare your data for use by other SAS Software procedures.

Core Concepts

Many SAS users work with data “collected in the time domain.” More formally, your data may be properly considered the “realization of a process occurring in the time domain,” and that the column vector of the realization “forms a time series.” In addition to the time-indexed column vector containing the realization, we often have a second vector which identifies the point in time at which the realization was measured. The second vector is often called the “index variable,” and in the SAS System, is a SAS date, time or datetime variable. This is a fancy way of saying that we have at least two variables in our data set, one which contains the values of some event of interest occurring in time and the second variable identifies the point in time at which the observation was taken.
**Forming a Time Series**

The values of a variable “form a time series” when they are ordered in ascending (lowest to highest) value by the index variable and there are no gaps between the values of the index variable. For example, if your series contains a realization of a process that occurs daily, then there must be an observation (row) in the data set for each day, even if on some days either you have missing data or the process of interest did not occur on a particular day. For example, suppose you have a time series with the number of vehicles which crossed a bridge on Mondays through Fridays for a year. No observations are available for Saturdays and Sundays. In order for your data to be properly considered a time series, observations for Saturdays and Sundays must be included in the data set. The values of the variable can be set to either zero or missing, as appropriate, for the weekend observations.

**Sampling Frequency**

The *sampling frequency* with which the process is measured is a critical concept. Simply put, the sampling frequency is how often the process is observed. So, your time series may have hourly, daily, monthly, quarterly observations. In some situations we may observe the process at random intervals, as in “surprise” or “spot-checks” to test, for example, the quality of a product or manufacturing process. Knowing the frequency of your series is important when deciding whether to *aggregate* or *interpolate* your observations to a *lower* or *higher* frequency.

**Aggregation to a Lower Sampling Frequency**

The process of “rolling-up” your higher sampling frequency data to a lower sampling frequency is called *aggregation*. If, for example, you create daily observations from a realization with an hourly sampling frequency, you are *aggregating* the series from a higher (hourly) to a lower (daily) frequency. PROC EXPAND is often a more effective method by which to aggregate data than a combination of Data Step processing followed by use of either PROC MEANS or PROC FREQ.

**Interpolation to a Higher Sampling Frequency**

Estimating higher sampling frequency observations from a time series realization collected at a lower sampling frequency is called *interpolation*. For example, estimating monthly observations from data collected at the quarterly sampling frequency is *interpolation*. PROC EXPAND, by default, applies a *cubic spline function* to the input time series to interpolate higher sampling frequency observations from processes observed at lower frequencies. PROC EXPAND is the only Procedure in the SAS System that can interpolate higher sampling frequency observations from lower sampling frequency observations.

**Interpolation of Missing Observations in a Time Series**

Missing observations in a time series pose many problems for the data analyst, statistician or econometrician. The forecasting procedures in SAS/ETS Software, such as PROC ARIMA and PROC FORECAST, require the input time series realization to be free of missing observations; it will have difficulty generating statistically valid forecasts from it. Even a simple data aggregation and reporting project can suffer from missing time series data. Suppose you need to “roll up” your monthly data to quarterly data for a reporting project. If a month’s data are missing, the resulting aggregation from monthly to quarterly data will mean that one quarter’s values will be based on two, rather than three months of data, leading to misleading and/or erroneous results.

Missing data pose many analytical challenges, and SAS Software contains several tools to replace missing values with some other value obtained, in some way, from the values of the non-missing observations. These tools include the REPLACE option in PROC STANDARD, a BASE SAS module procedure, and PROCs STDIZE, MI and MIANALYZE in SAS/STAT Software. The latter two Procedures, which were experimental in Version 8.2 of the SAS System, are production elements of SAS/STAT Software in SAS 9.

These tools are not, however, well-suited to handling missing data issues with data collected in the time domain. PROC EXPAND offers a superior set of tools for interpolating missing data from time series realizations than are provided in the other SAS Software procedures mentioned above because the interpolation features in PROC EXPAND can take in to account both seasonality and/or trend in a time series. The other missing data procedures/tools cannot do so.

**Default Interpolation Method in PROC EXPAND: The Cubic Spline Function**
In nautical terms, a “spline” is a knot that is tied to join two pieces of rope. Mathematically, a “spline function” joins two (or more) segments of a time series, resulting in a continuous time approximation across the entire series. By default, PROC EXPAND fits a cubic spline function to the data that is constrained to fit to the observed data points. This function is then used to obtain estimates of the missing values in the series. PROC EXPAND provides optional interpolation methods; please see the procedure documentation of details.

PROC EXPAND Syntax

The general form of a PROC EXPAND “step” is:

```
PROC EXPAND DATA=input_dsn OUT=output_dsn
   FROM=time_int TO=time_int METHOD=conversion_method;
   BY by_variable(s);
   CONVERT var = new_var/OBSERVED=frequency
   TRANSFORMIN = (transformation operators)
   TRANSFORMOUT = (transformation operators)
   OBSERVED= observational_characteristic;
   ID date_var;
RUN;
```

In the preceding example, optional statements, and arguments to the options are shown in italics. All non-italicized statements are required.

PROC EXPAND Statement

The PROC EXPAND Statement “calls” the Procedure. The DATA=input_dsn statement identifies the name of the input dataset while the OUT=output_dsn gives the name of the data set which will hold the procedure’s output. The FROM = time_int identifies the sampling frequency of the variable(s) in the input data set that will be placed in subsequent CONVERT statements for analysis by PROC EXPAND. The TO = time_int option supplies the sampling interval for the observations in the output data set given in the OUT=output_dsn option. If the TO = time_int option is not used observations in the output data set have the same sampling frequency as the observations in the input data set.

The BY Statement

PROC EXPAND supports optional by-group processing of observations in your data sets. Your data set must be sorted by the values of the BY variables, and within the BY variable values, in ascending order of the variable supplied in the required ID Statement (see below).

The CONVERT Statement

The CONVERT Statement identifies to PROC EXPAND the variable(s) upon which you want it to operate, the desired operations, and, optionally, the name of the variable in the output data set in which the desired operations will be stored.

Here are a few general comments about the CONVERT Statement:

- Only numeric variables can be analyzed by PROC EXPAND. Giving character variable names to a CONVERT Statement will generate an error message and cause PROC EXPAND to stop execution.
- A separate CONVERT Statement is required for each analysis/transformation to be performed by PROC EXPAND.
- Multiple CONVERT Statements can be placed in a single PROC EXPAND “step.”
- The same variable can be placed in multiple CONVERT Statements in the same PROC EXPAND “step.” This is useful when different transformation operators (see below) need to be applied to the same variable.
- Different variables in the same data set can be analyzed by PROC EXPAND in separate CONVERT Statements.
• Input data set variables not given in the **BY, CONVERT or ID** Statements are not copied to the output data set created by **PROC EXPAND**.

The **ID** Statement identifies to **PROC EXPAND** the input data set variable which is used as the time series index variable. **PROC EXPAND** assumes the variable given in the **ID** Statement is a SAS date, time or datetime variable. (Note: Observations in the input data set upon which **PROC EXPAND** is to be applied must be sorted in ascending order of the values of the variable placed in the **ID** Statement.)

Omitting the **ID** Statement will not cause **PROC EXPAND** to “error out,” but the values it generates and subsequently places in the output data set may be of very limited usefulness. In most situations where the **ID** Statement has been omitted **PROC EXPAND** will assume that the first observation in your data set has a SAS date value of 0 (zero), or January 1, 1960 and will then increment each succeeding observation in the output data set using the **INTNX** function based on the interval specified in the **FROM=** option in the **PROC EXPAND** Statement. If a time interval is not specified in a **FROM=** then **PROC EXPAND** will generate **ID** values as the observation count minus one.

The best way to avoid any problems with the output generated by **PROC EXPAND** when the **ID** Statement is omitted is use it in the first place. Values of the date, time or datetime variable given in the **ID** Statement will be used by **PROC EXPAND** to accurately account for calendar effects (e.g., the number of days in a month) when aggregating or interpolating your time series data.

**Example 1:** *Using PROC EXPAND to Interpolate Missing Values in a Time Series*

A permanent SAS data set **SASCLASS.CARMISS**, contains monthly counts of vehicles crossing the Golden Gate Bridge southbound from Marin County to San Francisco, California via the designated morning “rush hour” carpool lanes. Some months have missing values for the variable **CARS**. The variable **DATE** is a SAS date variable whose values are the first day of the month in which the observations were taken.

The following **PROC EXPAND** "step" will apply the default cubic spline function to the observations in **SASCLASS.CARMISS** and create a temporary output SAS data set, **NEWCARS**. The **FROM=** option informs **PROC EXPAND** that the data form a monthly time series. The **CONVERT** Statement instructs **PROC EXPAND** to create a new variable **NEWCARS** based on the values generated by applying the cubic spline function to the values of the variable **CARS**. Non-missing values of **CARS** in the input data set will be copied over to **NEWCARS** in the output data set. If a value of **CARS** in the input data set is missing, the value of **NEWCARS** in the output data set will contain the interpolated value, from fitting the cubic spline function to the non missing values of **CARS** in the series.

```sas
PROC EXPAND DATA=SASCLASS.CARMISS
   FROM=MONTH OUT=NEWCARS;
ID DATE;
CONVERT CARS=NEWCARS;
RUN;
```

Since only the **FROM=** option was used, the observations in the output SAS data **NEWCARS** are also monthly observations.

**Example 2:** *Aggregating Monthly Observations to Quarterly Observations with Missing Value Interpolation*

This **PROC EXPAND** example shows how the procedure is used to aggregate observations from a higher (in this case, monthly) frequency to a lower frequency (quarterly, in this example).

By default, **PROC EXPAND** will interpolate missing values in the input series before performing the requested aggregation or interpolation. So, in one “step” have the missing values in the monthly series from
SASCLASS.CARMISS replaced with estimated values and then the “cleaned up” series of monthly observations is “rolled up” to quarterly values in the temporary output data set CARS_QTR.

(In a subsequent example we will see how to apply options in PROC EXPAND that will suppress the default interpolation of any missing values prior to interpolation or aggregation.)

PROC EXPAND DATA=SASCLASS.CARMISS
   FROM=MONTH TO=QTR OUT=CARS_QTR;
ID DATE;
CONVERT CARS=CARS_QTR;
RUN;

Example 3A: Interpolating Weekly Estimates from Monthly Observations

Suppose an estimate of weekly carpool vehicle crossing is desired from the available monthly series. The desired estimated values are easily interpolated by PROC EXPAND in the next example and output to a new (temporary, in this example) data set.

PROC EXPAND DATA=SASCLASS.CARMISS
   OUT=WEEKLY1 FROM=MONTH TO=WEEK;
ID DATE;
CONVERT CARS=WEEKLY1;
RUN;

As with the previous examples, PROC EXPAND will first interpolate any missing values of CARS in the source data set and then interpolate the weekly values requested by specifying TO=WEEK. By default, the value of the index variable DATE in the output data set will be set equal to the first date in the interval. In this example, the first date in the WEEK interval corresponds to the value 1 returned by the WEEKDAY function, or Sunday.

Example 3B: Interpolating Weekly Estimates from Monthly Observations with a SAS Date Alignment Operator

In our analytic situation the carpool lanes are only open on weekdays (Mondays to Fridays), so perhaps we’d like PROC EXPAND to set the first day of each week interval to a Monday (or the value 2 returned by the WEEKDAY function). This can easily be accomplished by specifying a SAS date alignment operator in the TO= option. [SAS date alignment operators are discussed at length in the SAS Language: Concepts documentation.]

The PROC EXPAND task below carries out all the work specified in Example 3A. With the addition of a SAS date alignment operator in the TO= option, the observations in the output SAS data set have Mondays as their values, rather than the default Sunday values generated by the PROC EXPAND task shown in Example 3A.

PROC EXPAND DATA=SASCLASS.CARMISS FROM=MONTH TO=WEEK.2 OUT=WEEKLY2;
ID DATE;
CONVERT CARS=WEEKLY2;
RUN;

Specifying WEEK.2 as the argument to the TO= option instructs PROC EXPAND to create “weekly observations starting on Mondays” rather than the default Sundays.

Transformation Operators

A key addition to PROC EXPAND capabilities first added in Release 6.12 of the SAS System are a series of transformation operators which can be applied to your time series either before or after PROC EXPAND carries out the desired interpolation or aggregation of your time series. These operators can eliminate the need for complex Data Step treatment of observations in your data sets.
To fix ideas, let’s first look at applying a transformation operator to a time series before it is aggregated from a higher to a lower sampling frequency. Returning to the Golden Gate Bridge carpool data discussed earlier, suppose another data set is available that has a hourly count of carpool vehicles crossing in the southbound direction (that is, from Marin County to San Francisco) for just the days/hours that the designated carpool lanes are open: Monday through Friday, 5 AM to 10 AM. The data set has three years of hourly data, or 28,280 observations (24 hours times 365 days times 3 years), with observations for hours before/after the carpool lanes are open set to missing. Additionally, all observations for Saturdays, Sundays and holidays are set to missing.

What we want to do is aggregate the non-missing hourly observations to a monthly count of carpool vehicle crossings. By default, however, PROC EXPAND would fit a cubic spline function to all the observations in the series and from that function estimate values for the observations with missing hourly count values. For this analysis, we don’t want PROC EXPAND to do this, since all we want is a count of the actual recording carpool vehicle crossings “rolled up” from the observed hourly data to monthly observations.

One approach might be to use a Data Step to convert each missing hourly value from “missing” to a zero and then have PROC EXPAND perform the desired aggregation. While this approach would yield the desired results, we can save Data Step processing time (as well as programmer time required to write the code) by using the SETMISS transformation operator in the TRANSFORMIN option to the CONVERT statement. The TRANSFORMIN option supplies PROC EXPAND with a data transformation to be applied to the input series before aggregation or interpolation is carried out. In subsequent examples we will see how the TRANSFORMOUT option is used to apply transformation operators to a series after PROC EXPAND has carried out the desired aggregation or interpolation.

Example 4: Using the SETMISS Option to Supply a Replace Missing Values in a Time Series with a User-Defined Value.

In the following example, the hourly carpool vehicle crossing data set is aggregated to a monthly series. The SETMISS transformation operator is added to the TRANSFORMIN option in the CONVERT statement to replace all missing values of the analysis variable HOURLY_COUNT with a zero. Thus transformed, PROC EXPAND does not interpolate missing values in the series, since none exist for the purposes of the procedure’s operations upon the observations.

This example also implements the METHOD=NONE option, which suppresses PROC EXPAND’s fitting of a cubic spline function to the data (after the specified transformation). When there are no missing values in the series to be interpolated, using the METHOD=NONE option can save some processing time.

PROC EXPAND DATA=SASCLASS.HOURLY
   FROM=HOUR TO=MONTH OUT=MONTHLY_COUNT METHOD=NONE;
   ID DATE;
   CONVERT HOURLY_COUNT=MONTHLY_COUNT/
      TRANSFORMIN = (SETMISS 0);
RUN;

Moving Time Window Statistics

In SAS 9 Software PROC EXPAND provides over 90 transformation operators that can be used in the TRANSFORMIN and TRANSFORMOUT options. These operators are summarized in Table 14.1 in the SAS/ETS Documentation manual for SAS 9. (Similar tables are available in the SAS/ETS documentation for PROC EXPAND in Versions 6 and 8.)

Of these operators, perhaps the Moving Time Window Statistics are of the broadest general interest to SAS users. These operators compute measures such as moving sums, moving averages, moving ranges, and lags for a user-supplied number of time periods.
The usefulness of the *Moving Time Window Statistics* among the transformation operators should be immediately apparent to anyone who has had to use a Data Step to calculate these measures. Because the Data Step processes one observation at a time, a complex series of RETAIN statements are required to “hold” values of variables in the Program Data Vector from observation to observation. A simple end-user request to include the “six month rolling average” in a report that already includes the “three month rolling sum” might require extensive re-programming to accomplish.

Using PROC EXPAND, however, eliminates the need to write complex Data Steps to calculate these measures. If for no other reason, programmers who have no other need for SAS/ETS Software capabilities (e.g., forecasting or other modeling of time series data) would probably find that taking the time to master PROC EXPAND’s variable transformation capabilities will pay off in time no longer spent on tedious Data Step coding.

*Example 5: Using Moving Time Window Statistics in the TRANSFORMOUT Statement*

In this example several CONVERT Statements are used in a single PROC EXPAND task to calculate moving time window statistics of interest from the hourly Golden Gate Bridge carpool series. As with Example 4, above, the SETMISS transformation operator is supplied to the TRANSFORMIN option to set missing hourly observations to zero, thus overriding the procedure’s attempt to interpolate values for the hours/dates that the carpool lanes are closed. The observations are aggregated from hourly to weekly. After aggregation, various transformation operators are specified.

```
PROC EXPAND DATA=SASCLASS.HOURLY METHOD=NONE FROM=HOUR TO=WEEK.2;
ID DATE;
CONVERT HOURLY_COUNT = WEEKLY_COUNT /TRANSFORMIN=(SETMISS 0);
CONVERT HOURLY_COUNT = LAG1 / TRANSFORMIN=(SETMISS 0) TRANSFORMOUT = (LAG 1);
CONVERT HOURLY_COUNT = DIF1 / TRANSFORMIN=(SETMISS 0) TRANSFORMOUT = (DIF 1);
CONVERT HOURLY_COUNT = MOVESUM3 / TRANSFORMIN=(SETMISS 0) TRANSFORMOUT = (MOVSUM 3);
CONVERT HOURLY_COUNT = MOVEAVE3 /
TRANSFORMIN=(SETMISS 0) TRANSFORMOUT=(MOVAVE 3);
CONVERT HOURLY_COUNT = PCTDIF / TRANSFORMIN=(SETMISS 0) TRANSFORMOUT=(PCTDIF 1);
RUN;
```

In Example 5, multiple CONVERT statements were used to calculate several moving time window statistics from the same variable in the input SAS data set. The variables created are:

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>WEEKLY_COUNT</td>
<td>Simple sum of observations aggregated from HOUR to WEEK</td>
</tr>
<tr>
<td>LAG1</td>
<td>Lag 1 (the immediately preceding) observation from the aggregated series.</td>
</tr>
<tr>
<td>MOVESUM3</td>
<td>&quot;Rolling Three Period [week] Sum&quot; of the preceding three observations in the aggregated series</td>
</tr>
<tr>
<td>MOVEAVE3</td>
<td>&quot;Rolling Three Period [week] Average&quot; of the preceding three periods in the aggregated series</td>
</tr>
<tr>
<td>PCTDIF</td>
<td>Percentage change</td>
</tr>
</tbody>
</table>
Summary and Conclusions

PROC EXPAND is a very powerful tool with which you can rapidly and easily perform “housekeeping” operations on your time series data before applying other SAS Software tools to them. The wide array of transformation operators added to PROC EXPAND in Release 6.12, Version 8 and SAS 9 continue to extend the tools that the procedure offers to those users of SAS Software that work with data collected in the time domain.

Learning More

You can learn more about PROC EXPAND’s capabilities from the procedure’s chapter in the SAS/ETS documentation.

Author contact

Andrew H. Karp
President
Sierra Information Services, Inc.
19229 Sonoma Highway PMB 264
Sonoma, California  94115 USA
707 996 7380
SierralInfo@AOL.COM
www.SierralInformation.com

Copyright

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the United States of America and other countries. ® indicates USA registration. Other brand or product names are registered trademarks or trademarks of their respective companies.