A Macro to Associate Time-Oriented Data from Two Independent Sources.

David Arnold, Quintiles, Inc., RTP, NC

Abstract:

A macro is presented which automates the process of associating appropriate values from time-oriented data. Time-oriented data, or event data, represents conditions present at a given time, or conditions that continue over time. Often event data can come from independent sources, having been collected by separate methods, without common identifying time values. It may be necessary to describe the conditions in one source in the context of conditions present concurrently in another. The question is essentially, "what happened while a particular event occurred?" One accepted way of handling two independent sources of event data is to form a Cartesian product (map every event in A to every event in B), and then use logic to single out the appropriate associated events. This method is costly in terms of space and processing time. The macro performs the association without a Cartesian product, and thus is more efficient.

This macro might be useful in the analysis of data from clinical trials, basic biomedical research, or financial analysis. In particular, the macro could be used as a pre-processing tool for applications like SAS/PH-Clinical®, which make large working data sets from many data sources including event data.

Two hypothetical examples are given. The first relates dosing information collected in a pharmaceutical clinical trial with adverse experiences recorded while a patient is taking a test medication. The second demonstrates an association of financial data of with subsequent significant changes in the price of a company's stock.

Introduction:

Event data is a representation of conditions that occur in the context of time. While event data can be collected in a number of ways, we assume an organization of the data as follows: Each set of conditions that make up an event are represented by an observation in a SAS® data set. The macro is designed to associate event data that is in some way concurrent. For events to be concurrent, it is implied that at least one data source contains continuous event data, i.e. that conditions represented by an observation are presumed to persist until the next observation. Continuous event data should be organized accordingly, by recording conditions occurring at a given time point and then adding a new record to the data set when any of the conditions change. The unique identifier of any record is the date/time value for the event or change in condition, usually with a subject identifier. The macro deals with two events data sets at a time both of which have this type of organizational structure.

Terminology (Source and Target):

One data set is identified as the source, the other as the target. The source data set contains variables whose values will be carried over to the appropriate record in the target data set.

Assumptions:

1. Event data is continuous in at least the source data set. The conditions in the source are presumed to start at the date/time value occurring on a record, and are presumed to persist until the date/time
value of the next record when the records are taken in order.

2. The source data set must not have multiple observations with the same date/time value; these are presumed to be unique in the source.

3. Date/time values have the same measurement scale, and PROC SORT will order the observations correctly.

4. The variables to be passed from the source must be of the same type, character or numeric. If it is desirable to pass variables of both types the macro can be called twice once for the character variables and then subsequently for the numeric variables.

Requirements:

The macro should be as flexible as possible. The date/time variable, for example, could be days, seconds, alphanumeric, etc. The macro should allow the user to specify as a parameter the variables from the source data set to be included in the target data set. The macro should work in both directions, i.e. the user should be able to specify either the previous record or the next record from the source.

Examples:

Example 1: Passing the most recent values from the source data set

Consider as an example a hypothetical clinical trial of an experimental drug, designed to determine the maximum tolerated dose of the drug. The experience of certain adverse events were used as markers of the maximum tolerated dose. Participants in the trial began at a low dose of the drug, and at regular intervals the dose was escalated. Adverse events could occur at any time, and there could be some delay after a dose escalation before effects were seen. The data was collected in two time-oriented data sets. Dosing data was presumed to be continuous, that is, having been given a particular dose of medication, the participant continued at that dose until the next dose was specified. The dosing file had multiple records per patient. The first record contained the patient identifier, a visit number, the date of the initial dose and the strength of the medication. Subsequent records for that patient had the identifier, the visit number, the date the next dose was started and the strength. For the purposes of this example the adverse events are treated as discrete non-continuous events. The question of this hypothetical trial is at what dose did the subjects begin to experience significant side effects. Therefore, although the adverse events themselves have duration, the information that is important to our question is the time at which the adverse event was first reported. The adverse event file contained the following information: a patient identifier, the date of onset, the verbatim term the investigator used to describe the event, and certain attributes of the event, severity, whether hospitalization was required, whether the event could be related to the drug. The task for the analyst was to correctly associate the dose from the dosing file with each adverse event. The data for one patient is listed below in tables 1 and 2:

<table>
<thead>
<tr>
<th>Patient ID</th>
<th>Date</th>
<th>Visit</th>
<th>Strength</th>
<th>Treatment</th>
</tr>
</thead>
<tbody>
<tr>
<td>01JAN93</td>
<td>1</td>
<td>20 mg</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>01JAN93</td>
<td>2</td>
<td>25 mg</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>01FEB93</td>
<td>3</td>
<td>30 mg</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>01FEB93</td>
<td>4</td>
<td>35 mg</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>01MAR93</td>
<td>5</td>
<td>40 mg</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>01MAR93</td>
<td>6</td>
<td>45 mg</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>01APR93</td>
<td>7</td>
<td>0 mg</td>
<td>A</td>
<td></td>
</tr>
</tbody>
</table>
For the purposes of the macro the ID variable is PT, the "target" data set is the adverse events data set, and the "source" data is the dose data set. The variables to be carried over to the target are STRENGTH and TREATMNT. The macro call would be:

\%
carry(PT, dose, ac, dose date, onset, strength, treatment, character,);

Note that the default for the last parameter is to use the previous record from the source which is what we want in this example.

After the macro call the target data set includes STRENGTH and TREATMNT, shown in table 3:

<table>
<thead>
<tr>
<th>PT</th>
<th>ONSET</th>
<th>TERM</th>
<th>SEV</th>
<th>DOSE</th>
<th>RELATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>15JAN93</td>
<td>headache</td>
<td>mod.</td>
<td>20</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>15JAN93</td>
<td>nausea</td>
<td>mild</td>
<td>20</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>20FEB93</td>
<td>headache</td>
<td>mild</td>
<td>30</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>2FEB93</td>
<td>double vision</td>
<td>sev.</td>
<td>35</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>24FEB93</td>
<td>orthostatic changes</td>
<td>mod.</td>
<td>0</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>01APR93</td>
<td>car accident</td>
<td>mod.</td>
<td>0</td>
<td>A</td>
</tr>
</tbody>
</table>

If passing a numeric variable (VISIT) in addition to the two character variables was necessary, this could be accomplished with a second macro call:

\%
carry(PT, dose, ac, dose date, onset, visit, numeric,);

Example 2: Choosing the next observation from the source data set

In this example what we want is the next event in the source data set rather than the previous one. A financial analyst for XYZ Mutual Funds has the task of relating several indicators of the financial health of companies Z, Y, and X to fluctuations in stock prices. The analyst has in one data set the announced production level (thousands of units), number of orders shipped (thousands of units), and capital equipment purchased (thousands of dollars), and in another data set a record of significant changes in share price (> 5%). Both data sets are time-oriented, but events in each do not occur simultaneously. The records from each of the data sets for company X are shown in tables 4 and 5:

<table>
<thead>
<tr>
<th>COMPANY</th>
<th>RPT_DATE</th>
<th>PROD</th>
<th>SHIPPED</th>
<th>EQUIP</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>05APR92</td>
<td>255.0</td>
<td>132.1</td>
<td>0.0</td>
</tr>
<tr>
<td>X</td>
<td>11JUL92</td>
<td>330.4</td>
<td>256.2</td>
<td>255.7</td>
</tr>
<tr>
<td>X</td>
<td>03OCT92</td>
<td>175.0</td>
<td>220.3</td>
<td>400.1</td>
</tr>
<tr>
<td>X</td>
<td>06JAN93</td>
<td>357.9</td>
<td>323.5</td>
<td>31.2</td>
</tr>
</tbody>
</table>

The ID variable is COMPANY. The "target" data set is the INDICATR data set. The "source" data is the PRICE data set. The variables to be carried over to the target are PCT_CHNG and SHARE, numeric variables. The macro call would be:

\%
carry(company, price, indicatr, p_date, rpt_date, pctchg, share, numeric, next);

After the macro call the target data set includes PCT_CHNG and SHARE, shown in table 6:

<table>
<thead>
<tr>
<th>COMPANY</th>
<th>RPT_DATE</th>
<th>PROD</th>
<th>PCT_CHNG</th>
<th>SHARE</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>05APR92</td>
<td>255.0</td>
<td>6.7</td>
<td>12.57</td>
</tr>
<tr>
<td>X</td>
<td>11JUL92</td>
<td>330.4</td>
<td>6.7</td>
<td>12.57</td>
</tr>
<tr>
<td>X</td>
<td>03OCT92</td>
<td>175.0</td>
<td>-8.1</td>
<td>12.18</td>
</tr>
<tr>
<td>X</td>
<td>06JAN93</td>
<td>357.9</td>
<td>5.9</td>
<td>12.90</td>
</tr>
</tbody>
</table>
In this example it might have been useful to pass the date from the PRICE data set as well as PCT_CHNG and SHARE. Since the macro uses the date variable to sort and merge on, it is necessary to create a dummy date variable if we are going to pass it to the target as well:

data price;
  set price;
  dumbdate = p_date;
run;

%carry(company,price,indicatr,dumbdate,rpt_date,
  p_date pct_chng share,numeric,next);

Table 7 shows the resulting data set with the date from the PRICE data set.

Table 7  Indicator Data Set -- after macro processing
(not all variables are shown)

<table>
<thead>
<tr>
<th>COMPANY</th>
<th>RPT_DATE</th>
<th>P_DATE</th>
<th>PCT_CHNG</th>
<th>SHARE</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>05APR92</td>
<td>14JUL92</td>
<td>6.7</td>
<td>12.57</td>
</tr>
<tr>
<td>X</td>
<td>11JUL92</td>
<td>14JUL92</td>
<td>6.7</td>
<td>12.57</td>
</tr>
<tr>
<td>X</td>
<td>03OCT92</td>
<td>12DEC92</td>
<td>-6.1</td>
<td>12.18</td>
</tr>
<tr>
<td>X</td>
<td>04JAN93</td>
<td>11FEB93</td>
<td>5.9</td>
<td>12.90</td>
</tr>
</tbody>
</table>

Discussion/Conclusions

The macro uses a utility macro (SEPARATE) to identify the individual variable names passed in the &VARLIST parameter. The CARRY macro merges the source and target data sets on their respective date/times, then uses a retained buffer array (RETV AL(*)) to assign the appropriate values to the output target data set. The success of the macro processing is based on the sequential processing of data vectors in the data step, ordered by the event date/times from both data sources.

Attention should be given to sparse data occurring in the source. If multiple variables are to be passed to the target and some but not all are missing on any observation, unexpected results may occur. A non-missing value in the source is held in the retained buffer until the next non-missing value occurs.

In sparse data the retained values may accumulate, and eventually come from separate observations in the source. These values will be the most recent or next values relative to the observation in the target to which they are assigned but they may not have originated from the same observation. If this is not desirable, missing values in the source could be set to a non-missing value.

Processing of independent event-oriented data can potentially require a significant amount of space. Applications such as SAS/PH-Clinical often construct a meta-file representing the Cartesian product of the two source data sets in order to handle event data. Potentially the CARRY macro could be used to pre-process sources of event data, and thereby reduce the need for, or the size of a meta-file for this type of analysis.

Macro Code:

```
%macro separate(varlist);
%let num = 1;
%let varlist = &varlist;
%let varlen = length(&varlist);
%let start =
%do i = 1 %to &varlen;
  %let svar# = substr(&varlist,&start,%eval(&i-&start));
  %let start =
%end;
%global num;
%let num = &num + 1;
%macro separate(varlist);
%global num;
%let num = &num + 1;
%let varlist = &varlist;
%let varlen = length(&varlist);
%let start =
%do i = 1 %to &varlen;
  %let svar# = substr(&varlist,&start,%eval(&i-&start));
  %let start =
%end;
%global num;
%let num = &num + 1;
%end;
```

** CARRY.SAS -- The CARRY macro
** Written by: David Arnold, 2/22/93 rev. 9/1/93
********************************************************************************;
********************************************************************************;
** Macro SEPARATE:
** purpose : Separates a list of parameters
**            (variable names) passed as one character string into individual
**            parameters, maintaining their order and counting their number
** Written : 2/22/93, David Arnold
*******************************************************************************;
MACRO SEPARATE (varlist);
%global num;
%let varlist = &varlist;
%let varlen = length(&varlist);
%let start =
%do i = 1 %to &varlen;
  %if "substr(&varlist,i,1)" eq "" then %do;
    %let start = &i;
  %end;
%end;
```
**Macro CARRY**

**Purpose:** "Carry" over certain variables from one event data set (the "source") to another event data set (the "target") such that the values from the source are the most pertinent to the current observation in the target. The user specifies whether the most recent (previous) event or the next occurring event is to be the source observation for the specified variables. This macro is intended to be a more efficient way of linking two independent event data sets than the use of an SQL Cartesian product.

**Parameters:**
- **id**: ID variable common to source and target data sets.
- **source**: Source data set.
- **target**: Target data set.
- **s_date**: Source data set/time variable.
- **t_date**: Target data set/time variable.
- **varlist**: The list of variables of interest from the source (must be of one type, character or numeric).
- **type**: Type of variables in varlist (character or numeric).
- **next**: Next/previous, specifies whether the observation in the source data set with the closest date/time that is less than or equal to (next=previous) or greater than or equal to (next=next) the target date/time value.

**Requirements:**
1. **s_date** and **t_date** must be of the same type, both SAS date, time, or datetime values, or both a numeric or character scale that SAS is able to sort correctly with PROC SORT.
2. The variables in varlist must be of the same type, character or numeric. If it is desirable to pass variables of both types the macro can be called twice, once for the character var and then subsequently for the numeric var.

**Date:** 9/1/93

**Written by:** David Arnold,
Quintiles Inc.
P.O. Box 13979
RTP, NC 27709-3979

Macro carry(id,source,target,s_date,t_date, varlist,type,next);
Separate(varlist);
Ilist desc =;
If "next" eq "next" Ilist desc = descending;
proc sort data = source(keep = &id &s_date &varlist) out = source(rename = ( &s_date = eventdate )); by &id &desc &s_date;
run;
proc sort data = target out = target(rename = ( &t_date = eventdate ));
by &id &desc &t_date;
run;
If "type" = "numeric" Then;

**Data &target( rename = (eventdate = &t_date) drop = 1 vl-vnum);
merge &target(in = in_t) source;
by &id &desc eventdate;
retain vl-vnum;
aray retval(*) vl-vnum;
aray sources(*) &varlist;
if first.&id then do;
do i = 1 to &num;
retval(i) = . ;
end;
end;
do i = 1 to &num;
if sources(i) ne . then
retval(i) = sources(i);
*** map back the retained values ***;
sources(i) = retval(i);
end;
if in_t then output;
run;
Iend;

If."type" = "character" Then Id0;
*** Find lengths of source vars **************;
data _null_; set source;
retain vl-vnum 0;
aray widths(*) vl-vnum;
Id0 i = 1 It0 &num;
widths(i) = max( length(&&var&i), widths(i) );
call symput("wi", put(widths[i],3.));
Iend;
run;
data &target(rename = (eventdate = &t_date)
  drop = 1 vl-vnum);
Id0 i = 1 It0 &num;
length vl-w&i;
Iend;
merge &target(in = in_t) source;
by &id &desc eventdate;
retain vl-vnum "";
aray retval(*) vl-vnum;
aray sources(*) &varlist;
if first.&id then do;
do i = 1 to &num;
retval(i) = " ";
end;
end;
do i = 1 to &num;
if sources(i) ne "" then
retval(i) = sources(i);
*** map back the retained values ***;
sources(i) = retval(i);
end;
if in_t then output;
run;
Iend;

IIf "next" eq "next" Then Ido;
*** return target to an ascending order ***;
proc sort data = &target out = &target;
by &id &t_date;
run;
Iend;
Iend carry;
**END************CARRY Macro************;